Operations Research Letters 50 (2022) 356–361

Contents lists available at ScienceDirect

Operations Research Letters

www.elsevier.com/locate/orl

On the exactness of the ε-constraint method for biobjective nonlinear integer programming

Marianna De Santis [a,*], Gabriele Eichfelder [b], Daniele Patria [a]

[a] Dipartimento di Ingegneria Informatica Automatica e Gestionale, Sapienza Università di Roma, Via Ariosto, 25, 00185 Roma, Italy
[b] Institute for Mathematics, Technische Universität Ilmenau, Po 10 05 65, D-98684 Ilmenau, Germany

ARTICLE INFO

Article history:
Received 8 December 2021
Received in revised form 22 March 2022
Accepted 20 April 2022
Available online 26 April 2022

Keywords:
Multiobjective optimization
Integer programming
ε-constraint algorithm
Criterion space algorithms

ABSTRACT

The ε-constraint method is a well-known scalarization technique used for multiobjective optimization. We explore how to properly define the step size parameter of the method in order to guarantee its exactness when dealing with biobjective nonlinear integer problems. Under specific assumptions, we prove that the number of subproblems that the method needs to address to detect the complete Pareto front is finite. We report numerical results on portfolio optimization instances built on real-world data and show a comparison with an existing criterion space algorithm.

1. Introduction

There is a growing interest in devising exact methods for multiobjective integer programming (MOIP) as it is underlined by recent contributions in this respect (see, e.g. [2,6–9,21]). This is partly due to the fact that MOIPs represent a flexible tool to model real-world applications. Such models appear in works on finance, management, transportation, design of water distribution networks, biology [18,23–26]. MOIPs are intrinsically nonconvex, implying that the design of exact and efficient solution methods is particularly challenging and requires global optimization techniques [13]. In this paper, we focus on biobjective nonlinear integer programming problems of the following form

$$\begin{array}{ll} \min & (f_1(x), f_2(x))^T \\ \text{s.t.} & x \in \mathcal{X} \cap \mathbb{Z}^n, \end{array} \tag{BOIP}$$

where $\mathcal{X} \subseteq \mathbb{R}^n$ and $f_1, f_2 : \mathbb{R}^n \to \mathbb{R}$ are continuous functions. The image of the feasible set $\mathcal{X} \cap \mathbb{Z}^n$ under the vector-valued function $f : \mathbb{R}^n \to \mathbb{R}^2$ represents the feasible set in the *criterion space*, or the *image set*. When dealing with problem (BOIP), one wants to detect the so called *efficient solutions* $x^* \in \mathcal{X} \cap \mathbb{Z}^n$. Those are feasible points such that there exists no other feasible point $x \in \mathcal{X} \cap \mathbb{Z}^n$ for which $f_j(x) \le f_j(x^*)$, $j = 1, 2$ and $f(x) \ne f(x^*)$. The images $f(x)$ of efficient points $x \in \mathcal{X} \cap \mathbb{Z}^n$ are called *non-dominated points*. Furthermore, a point $\bar{x} \in \mathcal{X} \cap \mathbb{Z}^n$ is called a *weakly efficient* point of (BOIP) if there is no $x \in \mathcal{X} \cap \mathbb{Z}^n$ with $f(x) < f(\bar{x})$, where $<$ is meant componentwise. The images $f(x)$ of weakly efficient solutions $x \in \mathcal{X} \cap \mathbb{Z}^n$ are called *weakly non-dominated points*. We aim to solve (BOIP) exactly in the sense that we aim to detect its complete set of non-dominated points, also called the *non-dominated set* or *Pareto front*. In the following, we will denote the non-dominated set by $\mathcal{Y}_N$.

Regarding general purpose methods able to give correctness guarantees, the focus is most of all on multiobjective *linear* integer problems and we refer to [19] for a survey. A class of algorithms developed is the class of so called *criterion space search algorithms*, i.e., algorithms that work in the space of the objective functions (see, e.g. [16,21,22]). Such algorithms find non-dominated points by solving a sequence of single-objective linear integer programming problems. After computing a non-dominated point, these algorithms remove the dominated parts of the criterion space (based on the obtained non-dominated point) and look for not-yet-found non-dominated points in the remaining parts.

Criterion space algorithms usually rely on *scalarization* techniques. This means that the multiobjective problem is replaced with a parameter-dependent single-objective integer optimization problem. In order to find several non-dominated solutions, a sequence of single-objective integer optimization problems has to be handled considering different values of the parameters. A typical issue is how to choose the parameters so that the method can guarantee the detection of the complete non-dominated set. Criterion space algorithms have been extended to deal with nonlinear

* Corresponding author.
E-mail addresses: marianna.desantis@uniroma1.it (M. De Santis), gabriele.eichfelder@tu-ilmenau.de (G. Eichfelder), daniele.patria@uniroma1.it (D. Patria).

https://doi.org/10.1016/j.orl.2022.04.007
0167-6377/

problems, even if this clearly adds difficulties both from a theoretical and a numerical point of view. In case of (BOIPs), an approach that can be followed to deal with nonlinearities is the one proposed in [8]. There, the Frontier Partitioner Algorithm (FPA), relying on the use of the weighted-sum scalarization, has been proposed.

It is the purpose of this paper to use the ideas developed in [8] to define an exact criterion space algorithm based on another scalarization technique, the ε-constraint method. The ε-constraint method produces single-objective subproblems adding further constraints to the original feasible set. More specifically, given (BOIP), the ε-constraint method minimizes single-objective optimization problems of the following form:

$$\begin{array}{ll} \min & f_2(x) \\ \text{s.t.} & f_1(x) \leq \varepsilon \\ & x \in \mathcal{X} \cap \mathbb{Z}^n. \end{array} \tag{P_ε}$$

The parameter ε varies between $\min\{f_1(x) \mid x \in \mathcal{X} \cap \mathbb{Z}^n\}$ and $f_1(\hat{x}) - \delta$ with $\hat{x} \in \operatorname{argmin}\{f_2(x) \mid x \in \mathcal{X} \cap \mathbb{Z}^n\}$. Thereby, δ is a positive step size which influences the decrease of the upper bounds ε. As it will be clarified in the next section, the role of the step size δ is crucial to detect the complete non-dominated set of a (BOIP). The role of f_1 and f_2 in the definition of Problem (P_ε) can be interchanged. We refer to [5,12,17] for some examples on contributions on the ε-constraint method.

The paper is organized as follows. In Section 2, we recall the γ-positivity assumption introduced in [8] and we establish our main result. Under specific assumptions, we prove that the ε-constraint method is able to detect the complete Pareto front of a (BOIP) after having addressed a finite number of single-objective problems. In Section 3, we report our numerical experience. We compare the performance of FPA* [8], which is an improved version of FPA, with the ε-constraint method devised on portfolio optimization problems.

2. The γ-positivity assumption and the exactness of the ε-constraint method

The scheme of the ε-constraint method applied to (BOIP) is reported in Algorithm 1. As already mentioned, at every iteration k of the algorithm, the following single-objective integer subproblem needs to be addressed

$$\begin{array}{ll} \min & f_2(x) \\ \text{s.t.} & f_1(x) \leq \varepsilon^k \\ & x \in \mathcal{X} \cap \mathbb{Z}^n, \end{array} \tag{P_ε^k}$$

with $\varepsilon^k \in \mathbb{R}$ properly set. Recall that by [11, Proposition 4.3] any optimal solution of (P_ε^k) is a weakly efficient solution of (BOIP). Moreover, by [11, Theorem 4.5], for any efficient point x^* of (BOIP) there exists an upper bound $\varepsilon^k \in \mathbb{R}$ such that x^* is an optimal solution of (P_ε^k). However, we cannot solve (P_ε^k) for an infinite number of upper bounds ε^k. Thus, the question is whether and how we can find a finite number of upper bounds for which we have to solve (P_ε^k) such that we can still find all non-dominated points of (BOIP). In the following we discuss assumptions and an algorithm which guarantee such a finiteness result.

As a first assumption, we ask for the availability of a solver for (P_ε^k).

Assumption 2.1. There exists an oracle that either returns an optimal solution of (P_ε^k) or certifies its infeasibility for any choice of $\varepsilon^k \in \mathbb{R}$.

Note that there exists a number of solvers able to deal with single-objective nonlinear integer programming problems such as, e.g., BARON [20] or SCIP [14], so that Assumption 2.1 holds for many classes of (BOIP). In our computational experience, we will consider BOIPs having quadratic objective functions and a polyhedral set $\mathcal{X}$ and we will use GUROBI [15] as a solver.

Algorithm 1: Scheme of the ε-constraint method.

Input: (BOIP), $\delta > 0$, k = 1;
Output: the Pareto front $\mathcal{Y}_N$ of (BOIP);
Compute $x^* \in \operatorname{argmin}_{x \in \mathcal{X} \cap \mathbb{Z}^n} f_1(x)$
Compute $\hat{x} \in \operatorname{argmin}_{x \in \mathcal{X} \cap \mathbb{Z}^n} f_2(x)$
Set $\mathcal{M} = \{f(\hat{x})\}$
Set $\varepsilon^1 = f_1(\hat{x}) - \delta$
while $\varepsilon^k \geq f_1(x^*)$ **do**
  Compute $x^k \in \operatorname{argmin}_{x \in \mathcal{X}^k \cap \mathbb{Z}^n} f_2(x)$, with $\mathcal{X}^k = \mathcal{X} \cap \{x \in \mathbb{R}^n : f_1(x) \leq \varepsilon^k\}$
  Set $\mathcal{M} = \{f(x^k)\} \cup \mathcal{M}$
  Set $\varepsilon^{k+1} = f_1(x^k) - \delta$
  Set $k = k + 1$
end
Apply a filtering procedure to $\mathcal{M}$ to obtain $\mathcal{Y}_N$
Return $\mathcal{Y}_N$

In the definition of FPA (and FPA*) in [8] some basic assumptions on Problem (BOIP) had to be made. Here we make the same assumptions, reported in the following.

Assumption 2.2 *(Existence of the ideal point).* We assume that the ideal objective values $f_i^{\mathrm{id}} := \min_{\mathcal{X} \cap \mathbb{Z}^n} f_i(x)$, $i = 1, 2$, and thus the ideal point $f^{\mathrm{id}} := (f_1^{\mathrm{id}}, f_2^{\mathrm{id}}) \in \mathbb{R}^2$, exists.

The crucial assumption that we make in order to prove the exactness of the ε-constraint method is the so called *positive gap value* assumption. We need to assume that a positive value exists that underestimates the distance between the image of two integer feasible points of (BOIP), componentwise.

Definition 2.3 *(Positive γ-function).* Let $\gamma > 0$. A function $g : \mathcal{X} \to \mathbb{R}$ is a positive γ-function over $\mathcal{X} \cap \mathbb{Z}^n$ if it holds $|g(x) - g(z)| \geq \gamma$ for all $x, z \in \mathcal{X} \cap \mathbb{Z}^n$ with $g(x) \neq g(z)$.

Assumption 2.4. The functions $f_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, 2$ in Problem (BOIP) are positive γ-functions as in Definition 2.3 for some $\gamma > 0$.

Assumptions 2.2 and 2.4 imply that the non-dominated set $\mathcal{Y}_N$ of (BOIP) is finite (see Proposition 2.7 in [8]). Thus, we know that there exists a finite number of upper bounds ε^k for which we have to solve (P_ε^k) to find the complete Pareto front. The question is how to find this parameter set, and Algorithm 1 proposes an answer for that. Note that there exist a number of classes of functions that easily satisfy Assumption 2.1 and Assumption 2.4, such as linear or quadratic functions defined over $\mathbb{Q}^n$, or polynomials with rational coefficients over $\mathbb{Z}^n$ as long as they have no roots in $\mathbb{Z}^n$. Table 1, in Section 4.3 in [8], shows some classes of functions for which Assumption 2.4 holds and reports how to compute γ. The following example shows a case where only Assumption 2.2 holds, while Assumption 2.4 does not.

Example 2.5. Let $\mathcal{X} \cap \mathbb{Z} = \{x \in \mathbb{Z} \mid x \geq 0\}$, $f_1(x) = \arctan(x)$ and $f_2(x) = (x-1)^2$. We have that $f^{\mathrm{id}} = (0, 0)$, so that Assumption 2.2 is verified. However, it is not possible to find $\gamma > 0, \gamma \in \mathbb{R}$ such that $|\arctan(x) - \arctan(y)| \geq \gamma$ for all $x, y \in \mathcal{X} \cap \mathbb{Z}$ with $\arctan(x) \neq \arctan(y)$. Therefore, Assumption 2.4 does not hold.

We further want to underline that it is common to assume $\mathcal{X}$ to be bounded so that $\mathcal{X} \cap \mathbb{Z}^n$ and $f(\mathcal{X} \cap \mathbb{Z}^n)$ are finite sets. In this case, Assumption 2.4 trivially holds.

Remark 2.6. In the context of the ε-constrained method, Assumption 2.4 can be relaxed, requiring that only the function that is moved to the constraints (that is f_1 in (P_ε^k)) is a positive γ-function. Indeed, in all the proofs presented here, we need the positive γ-function property only for f_1. Moreover, since there are no two different non-dominated points with the same first component, the proof in [8, Proposition 2.7] can be adapted to show that the non-dominated set $\mathcal{Y}_N$ of (BOIP) is still finite.

Let Assumption 2.4 hold for f_1 and f_2 with $\gamma > 0$. Let $\delta > 0$ be the input parameter for Algorithm 1. Two different scenarios occur:

- $\delta > \gamma$: in this case the ε-constraint method could miss some points of the Pareto front $\mathcal{Y}_N$, since the step size δ may be wider than the distance between two non-dominated points;
- $\delta \le \gamma$: the ε-constraint algorithm is able to detect the complete Pareto front $\mathcal{Y}_N$ as shown in the following.

As already mentioned, according to [11, Proposition 4.3], any optimal solution of (P_ε^k) is a weakly efficient solution of (BOIP). For $\varepsilon^0 := f_1(\hat{x})$ the point $\hat{x}$ is an optimal solution of (P_ε^0) by construction and thus weakly efficient. Note that we cannot prove that a solution of (P_ε^k) is efficient, since a point $\tilde{x} \in \mathcal{X}^k \cap \mathbb{Z}^n$ may exist such that $f_1(\tilde{x}) < f_1(x^k)$ and $f_2(\tilde{x}) = f_2(x^k)$. Hence, before the filtering step, the set

$$\mathcal{M} = \{f(x^{k-1}), f(x^{k-2}), \dots, f(x^1), f(\hat{x})\}$$

may contain points which are just weakly non-dominated without being non-dominated. We will show in the next lemmata that it holds $\mathcal{Y}_N \subseteq \mathcal{M}$. The filtering step excludes those points $z \in \mathbb{R}^2$ from $\mathcal{M}$ for which there exists some $y \in \mathcal{M}$ with $y_i \le z_i$, $i = 1, 2$ and $y \neq z$. Such a filtering procedure is cheap thanks to the fact that the points are already sorted w.r.t. increasing first component and all points in $\mathcal{M}$ are already weakly non-dominated. In practice, we scroll the list $\mathcal{M}$ and, in case the distance with respect to the first component of two subsequent points is less than or equal to 10^{-5}, we remove the first point. For finite sets the domination property holds, i.e., $f(\mathcal{X} \cap \mathbb{Z}^n) \subseteq \mathcal{Y}_N + \mathbb{R}^2_+$. Hence, we can use the same argumentation as in [8, Proposition 2.7] and get that for any weakly non-dominated point z in $\mathcal{M}$ which is not also non-dominated, a point $y \in \mathcal{Y}_N \subseteq \mathcal{M}$ exists with $y \le z$, $y \neq z$. Thus, any point $z \in \mathcal{M}$ with $z \notin \mathcal{Y}_N$ is in fact filtered out. It remains to show that $\mathcal{Y}_N \subseteq \mathcal{M}$.

First, we prove that the ε-constraint method can find, at every iteration, a not-yet detected weakly efficient solution of (BOIP). We also show that no non-dominated point is missed in-between, where in-between refers to the sorting w.r.t. the first component, i.e. $f_1(x^{k-1}), f_1(x^{k-2}), \dots, f_1(x^1), f_1(\hat{x})$. For the following, recall that $x^0 := \hat{x}$ is weakly efficient and is the point detected at 'iteration' $k = 0$ to $\varepsilon^0 := f_1(\hat{x})$.

Lemma 2.7. *Let Assumption 2.4 hold with $\gamma > 0$. Assume that $\delta \le \gamma$ in Algorithm 1, $k \ge 1$, and that the point x^{k-1} is the point detected at iteration $k-1$. Then, at step k, Algorithm 1 finds a weakly non-dominated point $f(x^k)$ and no non-dominated point y with $y \neq f(x^k)$, $y \neq f(x^{k-1})$ and with $f_1(x^k) \le y_1 \le f_1(x^{k-1})$ exists.*

Proof. Let $x^k \in \operatorname{argmin}_{x \in \mathcal{X}^k \cap \mathbb{Z}^n} f_2(x)$ where $\mathcal{X}^k = \mathcal{X} \cap \{x \in \mathbb{R}^n : f_1(x) \le \varepsilon^k\}$ and with $\varepsilon^k = f_1(x^{k-1}) - \delta$. By [11, Proposition 4.3] the point x^k is, as well as x^{k-1}, a weakly efficient solution of (BOIP). Assume that there exists an efficient point $\tilde{x} \in \mathcal{X} \cap \mathbb{Z}^n$ with $f_1(x^k) \le f_1(\tilde{x}) \le f_1(x^{k-1})$. By Assumption 2.4 we have either $f_1(\tilde{x}) = f_1(x^{k-1})$ or $f_1(\tilde{x}) \le f_1(x^{k-1}) - \gamma$.

First, let $f_1(\tilde{x}) = f_1(x^{k-1})$. As x^{k-1} was obtained by solving ($\mathrm{P}_\varepsilon^{k-1}$) and as $\tilde{x}$ is feasible for this problem we have $f_2(\tilde{x}) \ge f_2(x^{k-1})$. As $\tilde{x}$ is efficient we derive $f(\tilde{x}) = f(x^{k-1})$. Second, let $f_1(\tilde{x}) \le f_1(x^{k-1}) - \gamma$. As $-\gamma \le -\delta$ it follows $f_1(\tilde{x}) \le f_1(x^{k-1}) - \delta$. Then $\tilde{x}$ is feasible for (P_ε^k) and as a consequence $f_2(x^k) \le f_2(\tilde{x})$. As $\tilde{x}$ is efficient and we have assumed that $f_1(x^k) \le f_1(\tilde{x})$ we derive $f(x^k) = f(\tilde{x})$. $\square$

As a consequence of Lemma 2.7, we have that $y \in \mathcal{M}$ for any non-dominated point $y \in \mathcal{Y}_N$ with $y_1 \in [f_1(x^{k-1}), f_1(\hat{x})]$. By the definition of $\hat{x}$ there is no $y \in \mathcal{Y}_N$ with $y_1 > f_1(\hat{x})$. Next we show that we miss no non-dominated point y with $y_1 < f_1(x^{k-1})$ by stopping the while loop based on $\varepsilon^k = f_1(x^{k-1}) - \delta < f_1(x^*)$, or in case we do not start the while loop at all, based on $\varepsilon^1 = f_1(\hat{x}) - \delta < f_1(x^*)$.

Lemma 2.8. *Let Assumption 2.4 hold with $\gamma > 0$. Assume that $\delta \le \gamma$ in Algorithm 1 and that the algorithm stopped at some iteration k. Then, there is no non-dominated point y with $y_1 < f_1(x^{k-1})$ in case $k \ge 2$, and with $y_1 < f_1(\hat{x})$ in case $k = 1$.*

Proof. First, we consider the case $k \ge 2$. Then the algorithm stopped due to $\varepsilon^k = f_1(x^{k-1}) - \delta < f_1(x^*)$. Assume y is a non-dominated point with $y_1 < f_1(x^{k-1})$. By Assumption 2.4 we have $y_1 \le f_1(x^{k-1}) - \gamma$ and hence $y_1 \le f_1(x^{k-1}) - \delta < f_1(x^*)$ which is a contradiction to the definition of x^*. Next, let $k = 1$, i.e., the while loop did not start at all due to $\varepsilon^1 = f_1(\hat{x}) - \delta < f_1(x^*)$. Assume y is a non-dominated point with $y_1 < f_1(\hat{x})$. Then we obtain with Assumption 2.4, as before, that $y_1 \le f_1(\hat{x}) - \gamma \le f_1(\hat{x}) - \delta < f_1(x^*)$, which is again a contradiction to the definition of x^*. $\square$

By Lemma 2.7 and Lemma 2.8 we have $\mathcal{Y}_N \subseteq \mathcal{M}$. However, there may exist weakly non-dominated points that cannot be found by our algorithm, as the following example shows:

Example 2.9. Let $\mathcal{X} \cap \mathbb{Z} = \{1, 2, 3, 4\}$, $f_1(x) = 5 - x$ and $f_2(x) = 1$ for all $x \in \mathcal{X} \cap \mathbb{Z}$. Thus we have $\gamma = 1$. All feasible points x are weakly efficient, but only $x = 4$ is efficient. We apply Algorithm 1 with $\delta = 0.5 \le \gamma$. Let $x^* = 4$, $\hat{x} = 1$ and thus $\varepsilon^0 = f_1(\hat{x}) = 4$. For $\varepsilon^1 = 3.5$ we may compute $x^1 = 4$ and the algorithm would stop without finding the remaining weakly efficient solutions.

On the other hand, it may happen that the ε-constraint method needs to compute all weakly non-dominated points before detecting the complete Pareto front.

Example 2.10. Let $\mathcal{X} = \{x \in \mathbb{R}^2 \mid 100x_2 \ge -x_1 + 100,\ 0 \le x_1 \le 100,\ 0 \le x_2 \le 1\}$ and let $f_1(x) = x_1$ and $f_2(x) = x_2$. Note that $\mathcal{Y}_N = \{(0, 1)^T, (100, 0)^T\}$. After having detected the non-dominated point $(100, 0)^T$, the ε-constraint method may need to address 100 more single-objective integer linear programming problems. Indeed, it may find the 99 weakly non-dominated points $(y_1, 1)^T$ with $y_1 = 99, 98, 97, \dots, 2, 1$ before detecting $(0, 1)^T$ and with this additional point the complete Pareto front.

It is important to note that the choice of δ has no impact on the number of iterations needed by the algorithm to stop. This means that using $\delta^1 \le \gamma$ or $\delta^2 \le \gamma$ with $0 < \delta^1 < \delta^2 \le \gamma$ leads to the same. So there is no need to find the largest possible value for γ but any γ for which Assumption 2.4 is satisfied will be enough. It is just important that δ is not chosen larger than any possible γ.

Lemma 2.11. *Let Assumption 2.4 hold with $\gamma > 0$. Assume that $0 < \delta^1 < \delta^2 \le \gamma$ and that the point x^{k-1} is the point obtained by solving ($\mathrm{P}_\varepsilon^{k-1}$). Then a point $\bar{x}$ is an optimal solution of (P_ε^k) with $\varepsilon^k =$*

$f_1(x^{k-1}) - \delta_1$ if and only if $\bar{x}$ is an optimal solution of (P_ε^k) with $\varepsilon^k = f_1(x^{k-1}) - \delta_2$.
Moreover, for any optimal solution $\bar{x}$ of (P_ε^k) with $\varepsilon^k = f_1(x^{k-1}) - \delta$ for some $\delta \in (0, \gamma]$ it holds $f_1(\bar{x}) \le f_1(x^{k-1}) - \gamma$.

Proof. First, let $\bar{x}$ be an optimal solution of (P_ε^k) with $\varepsilon^k = f_1(x^{k-1}) - \delta_2$. Then $\bar{x}$ is feasible for the problem with $\varepsilon^k = f_1(x^{k-1}) - \delta_1$. Assume $\bar{x}$ is not optimal for that problem. Then there exists x' feasible for the problem with $\varepsilon^k = f_1(x^{k-1}) - \delta_1$ with $f_2(x') < f_2(\bar{x})$. As $f_1(x') \le f_1(x^{k-1}) - \delta_1$ we have $f_1(x') < f_1(x^{k-1})$ and we get by Assumption 2.4 $f_1(x') \le f_1(x^{k-1}) - \gamma$. Thus $f_1(x') \le f_1(x^{k-1}) - \delta_2$ and x' is feasible for (P_ε^k) with $\varepsilon^k = f_1(x^{k-1}) - \delta_2$. Thus $f_2(x') \ge f_2(\bar{x})$ which is a contradiction.

Next, let $\bar{x}$ be an optimal solution of (P_ε^k) with $\varepsilon^k = f_1(x^{k-1}) - \delta_1$. Thus $f_1(\bar{x}) < f_1(x^{k-1})$ and we get by Assumption 2.4 $f_1(\bar{x}) \le f_1(x^{k-1}) - \gamma$. Thus $f_1(\bar{x}) \le f_1(x^{k-1}) - \delta_2$ and $\bar{x}$ is feasible for (P_ε^k) with $\varepsilon^k = f_1(x^{k-1}) - \delta_2$. As the feasible set for (P_ε^k) for δ_2 is a subset of the feasible set for δ_1, we obtain that $\bar{x}$ is also optimal for the problem with δ_2. This also shows that any optimal solution $\bar{x}$ of (P_ε^k) with $\varepsilon^k = f_1(x^{k-1}) - \delta$ and $\delta \in (0, \gamma]$ satisfies $f_1(\bar{x}) \le f_1(x^{k-1}) - \gamma$. $\square$

Based on the previous lemmata we are able to prove the following result.

Theorem 2.12. *Let Assumptions 2.1, 2.2 and 2.4 hold. Let $\delta \le \gamma$. Algorithm 1 finds the complete Pareto front $\mathcal{Y}_N$ of (BOIP) after having addressed a finite number of single-objective integer programs.*

Proof. By Lemma 2.7 and Lemma 2.8 we have $\mathcal{Y}_N \subseteq \mathcal{M}$ and after the filtering step, as discussed above, we obtain exactly the set $\mathcal{Y}_N$. Thanks to Assumption 2.4 we have that the `while` loop will take at most $m' = \lfloor (f_1(\hat{x}) - f_1(x^*)) / \delta \rfloor$ iterations. Based on Lemma 2.11 it makes no difference within the `while` loop how $\delta \in (0, \gamma]$ is chosen and thus the upper bound is $m = \lfloor (f_1(\hat{x}) - f_1(x^*)) / \gamma \rfloor$ iterations. Then, considering the two single-objective integer programs tackled at the beginning of Algorithm 1 for the computation of x^* and $\hat{x}$, the total number of single objective integer programs addressed by Algorithm 1 is $m + 2$. $\square$

Note that the generated set $\mathcal{M}$ contains weakly non-dominated points only and in each step of the `while` loop in Algorithm 1 a not-yet detected weakly non-dominated point of (BOIP) is found. For that reason, the `while` loop will take at most as many iterations as the number of weakly non-dominated points. However, we have no guarantee that the set of weakly non-dominated points of (BOIP) is finite. But using the same arguments as in the proof of [8, Proposition 2.7.] we have that the number of weakly non-dominated points y with $f_1(x^*) \le y_1 \le f_1(\hat{x})$ is finite and that thus the number of iterations is finite. This is another possibility to prove Theorem 2.12 above. Moreover, in case there are no weakly non-dominated points which are not at the same time non-dominated, we need to solve exactly $|\mathcal{Y}_N| + 1$ single-objective integer programs.

3. Numerical results

In our computational experiments, we consider bi-objective nonlinear integer instances arising from portfolio selection problems. Let $\mu \in \mathbb{R}^n$ be the expected return and $Q \in \mathbb{R}^{n \times n}$ be the covariance matrix with respect to a specific set of assets. We consider the following model

$$\begin{array}{ll} \min & (-\mu^T x,\ x^T Q x)^T \\ \text{s.t.} & a^T x \le b \\ & x \ge 0 \\ & x \in \mathbb{Z}^n, \end{array} \tag{1}$$

where the elements of $a \in \mathbb{R}^n$ are the prices of the financial securities, $b \in \mathbb{R}$ is the budget of the investor and the non-negativity constraint rules out short sales. The decision variable $x_i \in \mathbb{Z}$, $i = 1, \ldots, n$ stands for the amount of unit of a certain asset the investor is buying. Note that for this model Assumption 2.1 is satisfied, as the single-objective integer subproblem built from problem (1) is an integer quadratic problem that can be addressed by e.g. `GUROBI` [15].

As benchmark data set, we used historical real-data capital market indices from the Eurostoxx50 index that were used in [4, 3] and are publicy available at https://host.uniroma3.it/docenti/cesarone/DataSets.htm. This data set was used for solving a *Limited Asset Markowitz (LAM)* model. For each of the 48 stocks the authors obtained 264 weekly price data, adjusted for dividends, from Eurostoxx50 for the period from March 2003 to March 2008. Stocks with more than two consecutive missing values were disregarded. Logarithmic weekly returns, expected returns and covariance matrices were computed based on the period March 2003 to March 2007. By choosing stocks at random from the 48 available ones, we built portfolio optimization instances of different sizes with $\mu \in \mathbb{Q}^n$ and $Q \in \mathbb{Q}^{n \times n}$. We decided to generate 60 different instances by considering $n = 5, 10, 25, 30$ stocks. For every n, 15 different instances have been generated. Hence, we got the covariances matrices, the expected returns and the prices for every combination by picking the proper information from the files provided. As in [1], for every instance, we set $b = 10 \sum_{i=1}^n a_i$, representing the budget of the investor. We compare the ε-constraint method with FPA*. This is an improved version of FPA which uses the so called *custom weighted-sum* scalarization. It is able to detect the complete Pareto front after having solved $|\mathcal{Y}_N| + 2$ integer programs (see [8] for further details). In order to run FPA* and the exact version of the ε-constraint method, we need a proper value $\gamma > 0$ so that the γ-positivity assumption is satisfied. Therefore, we had to pre-process the data. We trimmed the number of decimal digits to four and multiplied the entries by 10^3. As a consequence, we obtained $\gamma \ge 0.1$. Note that, since the entries of Q and μ are in $\mathbb{Q}$, the value γ can be defined as $1/r$, where $r \in \mathbb{N}$ is the least common multiple of the denominators of the rational coefficients (see [8, Proposition 4.14]).

Both in the implementation of FPA* and of the ε-constraint method, we considered the linear objective $-\mu^T x$ as the function defining the additional constraint in the single-objective integer subproblems. Consequently, both FPA* and the ε-constraint method have to deal with a sequence of single-objective convex quadratic integer problems with linear constraints. In our Python implementation of the two algorithms, we used the MIQP solver of `GUROBI` [15]. All experiments have been executed on an Intel Core i5-6300U CPU running at 2.40 GHz and all running times were measured in cpu seconds.

In Table 1 and Table 2 we report, for each instance, the CPU time and the number of iterations ($\mathrm{it}_{\mathrm{FPA}^*}$ and $\mathrm{it}_{\mathrm{eps}}$) needed by FPA* and the ε-constraint method to detect the non-dominated set. Note that the total number of single-objective subproblems solved by FPA* and ε-constraint method is $\mathrm{it}_{\mathrm{FPA}^*} + 2$ and $\mathrm{it}_{\mathrm{eps}} + 2$, respectively. FPA* and the ε-constraint method have very similar performances, as the number of subproblem addressed by the two algorithms resulted to be very close in practice. However, the number of subproblems solved by the ε-constraint method can be larger than $|\mathcal{Y}_N| + 2$, which is the number of subproblems solved by FPA*.

We further compare FPA* and the ε-constraint algorithm using performance profiles as proposed by Dolan and Moré [10]. Given

Table 1
Results on instances with $n = 5$ and $n = 10$ variables.

Instance	$n = 5$				$n = 10$			
	FPA*	it_{FPA^*}	ε-const	it_{eps}	FPA*	it_{FPA^*}	ε-const	it_{eps}
p1	24.99	7381	24.60	7381	238.65	34890	234.75	34915
p2	6.69	1465	6.55	1467	78.24	10395	76.97	10390
p3	34.13	6103	32.89	6105	60.35	6047	60.55	6053
p4	1.89	402	1.81	402	47.17	6972	46.36	6978
p5	6.02	1208	5.86	1208	182.29	32894	180.58	32953
p6	3.09	709	3.03	709	50.24	6268	49.40	6272
p7	8.77	1676	8.56	1677	17.28	2372	17.17	2375
p8	4.61	845	4.20	846	230.24	23700	220.50	23714
p9	1.73	412	1.59	413	102.46	13544	96.96	13548
p10	2.09	428	1.95	430	93.25	8469	88.34	8471
p11	7.79	1547	7.29	1550	103.55	14987	98.22	14989
p12	4.44	837	4.13	840	69.71	7779	66.24	7784
p13	40.04	7257	37.41	7262	57.35	5057	54.45	5058
p14	5.45	1258	5.06	1259	305.46	26330	291.58	26339
p15	10.73	1869	10.00	1874	48.47	4990	46.50	4995

Table 2
Results on instances with $n = 25$ and $n = 30$ variables.

Instance	$n = 25$				$n = 30$			
	FPA*	it_{FPA^*}	ε-const	it_{eps}	FPA*	it_{FPA^*}	ε-const	it_{eps}
p1	1300.58	64896	1265.80	64978	10068.92	168679	10137.36	168963
p2	3111.57	91538	3145.79	91595	3684.09	57207	3682.69	57209
p3	974.07	77143	989.09	77136	5381.17	85015	5416.53	85033
p4	543.68	35835	549.17	35850	9712.74	134844	9906.55	134681
p5	807.29	46351	821.02	46353	5226.97	87750	5311.91	87784
p6	737.88	36444	747.85	36438	7284.13	111329	7465.47	111349
p7	1723.60	126491	1733.17	126556	7036.66	131154	7121.33	131180
p8	954.39	39077	948.63	39085	6923.09	112315	6908.47	112349
p9	899.33	38413	884.50	38411	9984.14	139199	9973.19	139205
p10	631.12	28747	619.97	28755	7683.15	120145	7682.36	120152
p11	2326.07	84459	2305.43	84429	7489.53	121898	7480.73	121920
p12	1882.48	66347	1864.43	66359	4452.29	68145	4396.06	68150
p13	2234.69	105386	2201.69	105393	7281.69	128790	7164.96	128912
p14	1551.69	88263	1539.97	88210	4510.21	67305	4500.24	67342
p15	2224.76	112773	2182.22	112685	5657.23	87619	5647.94	87588

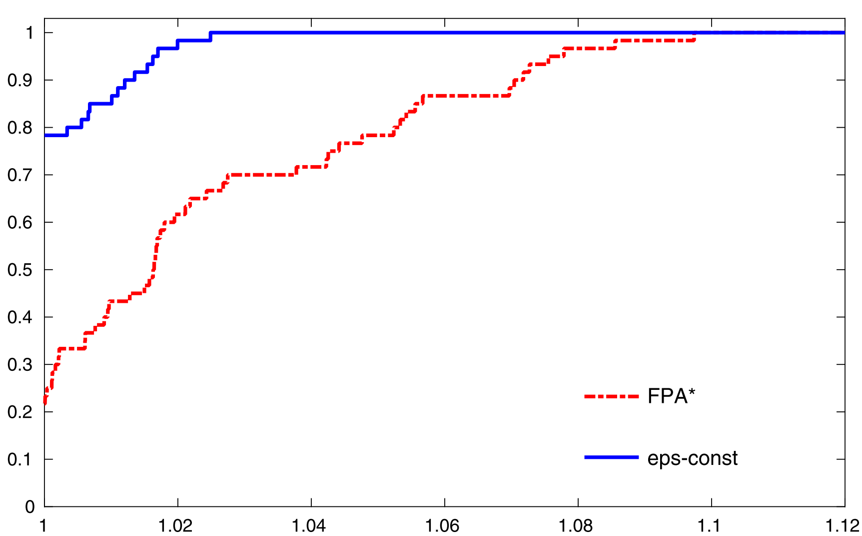

Fig. 1. Comparison between FPA* and the ε-constraint method on all the 60 instances.

a set of solvers $\mathcal{S}$ and a set of problems $\mathcal{P}$, the performance of a solver $s \in \mathcal{S}$ on problem $p \in \mathcal{P}$ is compared against the best performance obtained by any solver in $\mathcal{S}$ on the same problem. The performance ratio is defined as $r_{p,s} = t_{p,s}/\min\{t_{p,s'} \mid s' \in \mathcal{S}\}$, where $t_{p,s}$ is the measure we want to compare, and we consider a cumulative distribution function $\rho_s(\tau) = |\{p \in \mathcal{P} \mid r_{p,s} \leq \tau\}|/|\mathcal{P}|$. The performance profile for $s \in S$ is the plot of the function ρ_s. We report in Fig. 1 the performance profiles of FPA* and the ε-constraint algorithm with respect to the CPU time considering all the 60 instances. Note that the value τ needed to have both $\rho_{\varepsilon\text{-const}}(\tau) = 1$ and $\rho_{FPA^*}(\tau) = 1$ is very small ($\tau = 1.116$), confirming that the two algorithm share very similar performance.

Data availability

Data will be made available on request.

Acknowledgements

The first author acknowledges support within the project No RP1181641D22304F which has received funding from Sapienza University of Rome.

References

[1] C. Buchheim, M. De Santis, F. Rinaldi, L. Trieu, A Frank–Wolfe based branch-and-bound algorithm for mean-risk optimization, J. Glob. Optim. 70 (3) (2018) 625–644.
[2] G. Cabrera-Guerrero, M. Ehrgott, A.J. Mason, A. Raith, Bi-objective optimisation over a set of convex sub-problems, Ann. Oper. Res. (2021) 1–26.
[3] F. Cesarone, F. Tardella, Equal risk bounding is better than risk parity for portfolio selection, J. Glob. Optim. 68 (2) (2017) 439–461.
[4] F. Cesarone, A. Scozzari, F. Tardella, A new method for mean-variance portfolio optimization with cardinality constraints, Ann. Oper. Res. 205 (1) (2013) 213–234.
[5] K. Chircop, D. Zammit-Mangion, On ε-constraint based methods for the generation of Pareto frontiers, J. Mech. Eng. Autom. 3 (5) (2013) 279–289.
[6] M. De Santis, G. Eichfelder, A decision space algorithm for multiobjective convex quadratic integer optimization, Comput. Oper. Res. 134 (2021) 105396.
[7] M. De Santis, G. Eichfelder, J. Niebling, S. Rocktäschel, Solving multiobjective mixed integer convex optimization problems, SIAM J. Optim. 30 (4) (2020) 3122–3145.
[8] M. De Santis, G. Grani, L. Palagi, Branching with hyperplanes in the criterion space: the frontier partitioner algorithm for biobjective integer programming, Eur. J. Oper. Res. 283 (1) (2020) 57–69.
[9] S.F. Doğan, O. Karsu, F. Ulus, An exact algorithm for biobjective integer programming problems, Comput. Oper. Res. 132 (2021) 105298, https://doi.org/10.1016/j.cor.2021.105298.
[10] E.D. Dolan, J.J. Moré, Benchmarking optimization software with performance profiles, Math. Program. 91 (2) (2002) 201–213, https://doi.org/10.1007/s101070100263.
[11] M. Ehrgott, Multicriteria Optimization, Springer-Verlag Berlin Heidelberg, 2005.
[12] G. Eichfelder, Scalarizations for adaptively solving multi-objective optimization problems, Comput. Optim. Appl. 44 (2009) 249–273.
[13] G. Eichfelder, Twenty years of continuous multiobjective optimization in the twenty-first century, EURO J. Comput. Optim. 9 (2021) 100014, https://doi.org/10.1016/j.ejco.2021.100014.
[14] G. Gamrath, D. Anderson, K. Bestuzheva, W.-K. Chen, L. Eifler, M. Gasse, P. Gemander, A. Gleixner, L. Gottwald, K. Halbig, G. Hendel, C. Hojny, T. Koch, P. Le Bodic, S.J. Maher, F. Matter, M. Miltenberger, E. Mühmer, B. Müller, M.E. Pfetsch, F. Schlösser, F. Serrano, Y. Shinano, C. Tawfik, S. Vigerske, F. Wegscheider, D. Weninger, J. Witzig, The SCIP Optimization Suite 7.0, Tech. Rep., Optimization Online, 2020, http://www.optimization-online.org/DB_HTML/2020/03/7705.html.
[15] L. Gurobi Optimization, Gurobi optimizer reference manual http://www.gurobi.com, 2020.
[16] T. Holzmann, J.C. Smith, Solving discrete multi-objective optimization problems using modified augmented weighted Tchebychev scalarizations, Eur. J. Oper. Res. 271 (2) (2018) 436–449.
[17] M. Laumanns, L. Thiele, E. Zitzler, An efficient, adaptive parameter variation scheme for metaheuristics based on the epsilon-constraint method, Eur. J. Oper. Res. 169 (3) (2006) 932–942.
[18] A. Legendre, E. Angel, F. Tahi, Bi-objective integer programming for RNA secondary structure prediction with pseudoknots, BMC Bioinform. 19 (13) (2018).
[19] A. Przybylski, X. Gandibleux, Multi-objective branch and bound, Eur. J. Oper. Res. 260 (3) (2017) 856–872.
[20] N.V. Sahinidis, BARON: a general purpose global optimization software package, J. Glob. Optim. 8 (2) (1996) 201–205.
[21] S. Tamby, D. Vanderpooten, Enumeration of the nondominated set of multiobjective discrete optimization problems, INFORMS J. Comput. 33 (1) (2021) 72–85.
[22] O. Turgut, E. Dalkiran, A.E. Murat, An exact parallel objective space decomposition algorithm for solving multi-objective integer programming problems, J. Glob. Optim. 75 (1) (2019) 35–62.
[23] A.-J. Ulusoy, F. Pecci, I. Stoianov, Bi-objective design-for-control of water distribution networks with global bounds, Optim. Eng. (2021) 1–51.
[24] P. Xidonas, G. Mavrotas, J. Psarras, Equity portfolio construction and selection using multiobjective mathematical programming, J. Glob. Optim. 47 (2) (2010) 185–209.
[25] M.M. Yenisey, B. Yagmahan, Multi-objective permutation flow shop scheduling problem: literature review, classification and current trends, Omega 45 (2014) 119–135.
[26] H. Zhong, W. Guan, W. Zhang, S. Jiang, L. Fan, A bi-objective integer programming model for partly-restricted flight departure scheduling, PLoS ONE 13 (5) (2018) e0196146.