Check for
updates

Nonconvex constrained optimization by a filtering branch and bound

Gabriele Eichfelder[1] (ID) · **Kathrin Klamroth[2]** (ID) · **Julia Niebling[1]** (ID)

Abstract

A major difficulty in optimization with nonconvex constraints is to find feasible solutions. As simple examples show, the αBB-algorithm for single-objective optimization may fail to compute feasible solutions even though this algorithm is a popular method in global optimization. In this work, we introduce a filtering approach motivated by a multiobjective reformulation of the constrained optimization problem. Moreover, the multiobjective reformulation enables to identify the trade-off between constraint satisfaction and objective value which is also reflected in the quality guarantee. Numerical tests validate that we indeed can find feasible and often optimal solutions where the classical single-objective αBB method fails, i.e., it terminates without ever finding a feasible solution.

Keywords Constrained optimization · Nonconvex optimization · Global optimization · Branch and bound · Multiobjective optimization

Mathematics Subject Classification 90C26 · 90C29 · 90C30

1 Introduction and motivation

Numerical methods for constrained optimization problems have a multitude of applications in a large range of technical and economical applications. To give a concrete example, we consider an engineering design problem where the overall aim is to minimize the building costs while certain requirements on the quality have to be met. Depending on the structure

✉ Julia Niebling
   julia.niebling@tu-ilmenau.de

   Gabriele Eichfelder
   gabriele.eichfelder@tu-ilmenau.de

   Kathrin Klamroth
   klamroth@math.uni-wuppertal.de

[1] Institute for Mathematics, Technische Universität Ilmenau, P.O. Box 10 05 65, 98684 Ilmenau, Germany

[2] Department of Mathematics and Computer Science, University of Wuppertal, Gaußstr. 20, 42119 Wuppertal, Germany

of the problem at hand, it may already be a challenging task to find feasible solutions, and even more so to find feasible solutions that are provably optimal.

1.1 Constraint handling and multiobjective counterparts

In this paper, we focus on constrained optimization problems where the objective function as well as the constraints are given by twice continuously differentiable functions. Noting that feasibility constraints are a prevalent difficulty in constrained optimization, we take a multiobjective perspective and suggest to relax all complicating constraints and re-interpret them as additional and independent objective functions. From a practical point of view, such *multiobjective counterpart* models account for the fact that the right-hand-side values of constraints are often based on individual preferences and/or estimations (consider, for example, quality constraints in an engineering design problem). Indeed, slight constraint violations may be acceptable if this allows for a significant improvement of the primary (cost) objective function. Conversely, if a significantly better quality can be obtained at only slightly higher cost, the decision maker may be willing to invest more for a solution that in a sense "oversatisfies" the constraints. A multiobjective counterpart model provides trade-off information between constraint satisfaction on one hand and primary objective function on the other hand, and thus supports the decision making process and the selection of a most preferred solution.

From a numerical point of view, multiobjective counterpart models have another important advantage: By relaxing all complicating constraints, feasible solutions are easily available to initiate the optimization procedure. Depending on the selected constrained programming solver, this may actually be a crucial property, see Example 1 below. Moreover, even when the original constrained optimization problem is infeasible, multiobjective counterpart models have the potential to efficiently compute "best possible" solutions that, naturally, remain infeasible for the original constrained problem, but perform as well as possible w.r.t. both constraint satisfaction and primary objective in a multiobjective sense.

1.2 αBB method

The αBB-method suggested by [1] is an example of a deterministic constrained programming algorithm that aims at the determination of a globally optimal solution. It can be interpreted as a geometric branch-and-bound algorithm that discards subregions of the feasible set (referred to as *boxes*) based on efficiently computed lower and upper bounds on the optimal objective value. It terminates with the best found feasible solution as soon as the difference between upper and lower bound falls below a prespecified accuracy requirement. On a specific subbox, a lower bound is obtained by solving an auxiliary convex optimization problem (referred to as *convexified problem* in the following) restricted to a convex superset of the feasible set in the considered subbox. The objective function of this convexified problem is a convex underestimator of the original objective function, and thus an optimal solution yields a lower bound on the best possible objective value in the subbox in question. Upper bounds, on the other hand, are obtained from images of already known feasible solutions, and thus crucially depend on the ability to efficiently compute such feasible solutions. Usually, feasible solutions are retrieved from the optimal solutions of the convexified problems, i.e., by evaluating the respective solutions with the original constraint functions. However, since the convexified problems operate on relaxed (convex) feasible sets, their solutions do not have to be feasible for the original problem, and in this case the determination of upper bounds becomes

complicated. This difficulty was described in [17] and exemplified at the following example problem:

Example 1 [17] Consider the single-objective optimization problem

$$
\begin{aligned}
\min \ & f(x) = x_1 - x_2 \\
\text{s.t.} \ & g(x) = -x_1^2 - (x_2 - 5)^2 + 25 + \sqrt{2} \leq 0, \\
& x \in [1, 2] \times [0, 1].
\end{aligned}
\tag{1}
$$

The global optimum of problem (1) is $x^* = (\sqrt[4]{2}; 0) \approx (1.1892; 0)$, and the optimal objective value can be obtained as $f(x^*) = \sqrt[4]{2} \approx 1.1892$.

As was noted in [17], the αBB method does not make any progress when applied to problem (1) since the optimal solutions of the convexified problems are always infeasible for the original problem. As a consequence, lower bounds are constructed, while upper bounds are never obtained. Therefore, the αBB method can never satisfy the accuracy requirements and thus does not terminate. In the following, we will use Example 1 as a test case to exemplify the advantages and also the difficulties when considering multiobjective counterpart models and, as a natural consequence, a multiobjective counterpart of the αBB method that is specifically tailored for the solution of constrained optimization problems.

1.3 Related literature

The close relationship between constrained optimization problems and multiobjective counterpart models has been discussed and exploited in a variety of different ways. We refer to [18] for a general survey on the interrelation between different relaxation strategies of constrained programming problems (like, for example, Lagrangian relaxation and exact penalty functions) and corresponding scalarizations of the multiobjective counterpart model (in case of Lagrangian relaxation this is the classical weighted-sum scalarization, while exact penalty functions correspond to so-called elastic constraint scalarizations). In [15], a multiobjective counterpart approach is introduced for constrained multiobjective optimization problems and the interrelation of constrained and unconstrained multiobjective optimization is examined. Multiobjective counterpart models have motivated a variety of solution approaches for different classes of constrained optimization problems. As a prominent example, multiobjective counterpart models naturally relate to filter methods where total constraint violation is interpreted as a second objective function, see, for example, [11] and the survey in [12]. Multiobjective counterpart models are also used, among others, to handle constraints in evolutionary algorithms (see, for example, [32] for a survey), and in the context of combinatorial optimization problems to efficiently compute solution alternatives for multiobjective and multidimensional knapsack problems [30]. Finally, we note that besides constraint handling multiobjective optimization is also a powerful tool for other algorithmic aspects; see [34] for a recent example.

Common algorithms to solve multiobjective optimization problems globally can be divided into deterministic and stochastic algorithms. Stochastic algorithms, also named evolutionary methods, construct a population of (feasible) solutions and use evolutionary techniques in order to find new feasible solutions with a better objective value. See [3,4,13] for some exemplary procedures. A drawback of such algorithms is that they cannot guarantee to find optimal solutions in a finite amount of time.

Hence, we want to make use of a deterministic algorithm. Good surveys on deterministic methods in multiobjective optimization can be found in the books [22,25]. Most of the

known deterministic methods for biobjective optimization are based on the branch-and-bound approach, see, for example, [10,21,35]. The upper and lower bounds for "optimal" points in the image space are often obtained by using interval methods or Lipschitz properties of the objectives. For multiobjective optimization problems (with more than two objectives) only a few branch-and-bound algorithm exist, see [2,9,24,28,29]. In [24] an improved lower bounding procedure is introduced which uses αBB underestimators and Benson's outer approximation algorithm for convex multiobjective optimization problems, see [7,19]. Furthermore, the proposed algorithm guarantees a certain accuracy of the computed solutions in the pre-image and in the image space.

1.4 Contribution

It is well known from the literature that using multiobjective counterparts can be highly beneficial as outlined above. In a short conference proceeding, see [8], it was already mentioned by the authors of this paper that solving the nonconvex multiobjective counterpart by a global solution technique can deliver feasible solutions for the original constrained single-objective problem, and first numerical experiments on problem (1) were performed.

In this work, after introducing the basic notations and definitions in Sect. 2, we examine the relations between constrained optimization problems and their multiobjective counterparts in detail, see Sect. 3. We also study ε-optimality as well as weak optimality notions. In Sect. 4, we present for the first time a branch-and-bound based algorithm for the specific nonconvex multiobjective counterpart problem. This algorithm is specifically tailored to find those points which deliver useful information for the original constrained optimization problem—without waisting time by a full determination of the nondominated set of the multiobjective counterpart. We prove the finiteness and the correctness of the proposed new algorithm.

In Sect. 5, we propose some possible post-processing for improving the results further. Finally, we illustrate the new algorithm on various test instances in Sect. 6. We conclude with Sect. 7.

2 Definitions and notations

In this paper, we focus on two variants of *constrained optimization* problems. Let $X \subseteq \mathbb{R}^n$ be a *box*, i.e., there are vectors $\underline{x}, \overline{x} \in \mathbb{R}^n$ with $\underline{x} \leq \overline{x}$ and $X = \{x \in \mathbb{R}^n \mid \underline{x} \leq x \leq \overline{x}\}$. The inequality sign $\leq$ has to be understood component-wise. Furthermore, let $f : \mathbb{R}^n \to \mathbb{R}$ and $g_k : \mathbb{R}^n \to \mathbb{R}$, $k = 1, \ldots, r$ be twice continuously differentiable functions. For an arbitrary set $A \subseteq \mathbb{R}^n$, we denote the image set of A under the map f by $f(A) := \{f(x) \in \mathbb{R} \mid x \in A\}$. The following two constrained optimization problems are equivalent in the sense that they have the same feasible set and objective functions.

$$\min_{x \in X} f(x) \qquad \qquad (P_1)$$
$$\text{s.t. } g_k(x) \leq 0, \ k = 1, \ldots, r$$

$$\min_{x \in X} f(x) \qquad \qquad (P_2)$$
$$\text{s.t. } \max_{k=1,\ldots,r} g_k(x) \leq 0$$

We assume in the following that the feasible set S of (P_1) and (P_2) given by $S := \{x \in X \mid g_k(x) \leq 0, k = 1, \ldots, r\}$ is not empty. We denote the elements of the feasible set of the single-objective optimization problem by *feasible solutions*.

Definition 1 Let $\varepsilon > 0$ be given.

(i) A feasible solution $\bar{x} \in S$ is said to be minimal for (P_1) (or (P_2), respectively) if $f(\bar{x}) \leq f(x)$ for all $x \in S$.
(ii) A feasible solution $\bar{x} \in S$ is said to be ε-minimal for (P_1) (or (P_2), respectively) if $f(\bar{x}) < f(x) + \varepsilon$ for all $x \in S$.

Next, we formulate two box constrained multiobjective optimization problems that relax the constraints of (P_1) and (P_2), respectively, and reinterpret them as additional objective functions. We refer to these problems as the *multiobjective counterparts* to (P_1) and (P_2), respectively. The relations between the original problems and the counterpart problems are explored later.

$$\min_{x \in X} \begin{pmatrix} f(x) \\ g_1(x) \\ \vdots \\ g_r(x) \end{pmatrix} \qquad (MOP_1)$$

$$\min_{x \in X} \begin{pmatrix} f(x) \\ \max_{k=1,\ldots,r} g_k(x) \end{pmatrix} \qquad (MOP_2)$$

Note that the bi-objective problem (MOP_2) has an objective function which is in general not differentiable. To simplify the notation, we define the mapping $G \colon \mathbb{R}^n \to \mathbb{R}$ by

$$G(x) := \max_{k=1,\ldots,r} g_k(x).$$

Moreover, we write $(f, g_1, \ldots, g_r)(x)$ and $(f, G)(x)$ to denote the objective vector of a *solution* $x \in X$ w.r.t. problem (MOP_1) and (MOP_2), respectively.

2.1 Multiobjective counterpart models

Multiobjective counterpart models fall in the class of multiobjective optimization problems. In this context, optimality concepts are based on comparing outcome vectors subject to order relations. We use the classical concept of Pareto dominance that is based on the following order relation for two vectors $x, y \in \mathbb{R}^m$ (see, for example, [6] or [22]):

$$x \leq y \Leftrightarrow x_j \leq y_j \text{ for all } j = 1, \ldots, m$$
$$x \lneq y \Leftrightarrow x_j \leq y_j \text{ for all } j = 1, \ldots, m \text{ and } x \neq y$$
$$x \nleq y \Leftrightarrow x_j > y_j \text{ for at least one } j \in \{1, \ldots, m\}$$
$$x < y \Leftrightarrow x_j < y_j \text{ for all } j = 1, \ldots, m$$

The symbols $\geq, \gneq, \ngeq$ and $>$ are defined analogously.

To simplify notation, we denote the vector-valued objective functions of (MOP_1) and (MOP_2) by $h \colon \mathbb{R}^n \to \mathbb{R}^{1+r}$ with $h(x) = (f, g_1, \ldots, g_r)(x)$ and $h \colon \mathbb{R}^n \to \mathbb{R}^2$ with $h(x) =$

$(f, G)(x)$, respectively. Moreover, $e = (1, \dots, 1)^T \in \mathbb{R}^m$ is the m-dimensional all-ones vector, where $m = 1 + r$ or $m = 2$, respectively, depending on the dimension of the currently considered problem.

Remark 1 Note that the pair $((P_2), (MOP_2))$ is a special case of the pair $((P_1), (MOP_1))$ (with $r = 1$ and $g_1(x) = G(x)$). The following definitions and relations will be formulated for $((P_1), (MOP_1))$. They hold equivalently for $((P_2), (MOP_2))$.

Definition 2 Let $\varepsilon > 0$ be given.

(i) A solution $\bar{x} \in X$ is said to be *efficient* for (MOP_1) if there does not exist another solution $x \in X$ with $h(x) \leq h(\bar{x})$. If $\bar{x}$ is efficient, then the corresponding outcome vector $h(\bar{x})$ is called *nondominated*.

(ii) A solution $\bar{x} \in X$ is said to be *weakly efficient* for (MOP_1) if there does not exist another solution $x \in X$ with $h(x) < h(\bar{x})$. If $\bar{x}$ is weakly efficient, then the corresponding outcome vector $h(\bar{x})$ is called *weakly nondominated*.

(iii) A solution $\bar{x} \in X$ is said to be *ε-efficient* for (MOP_1) if there does not exist another solution $x \in X$ with $h(x) \leq h(\bar{x}) - \varepsilon e$.

Note that for all considered optimization problems (P_1), (P_2), and (MOP_1), (MOP_2), an optimal or efficient solution exists, respectively, because both feasible sets S and X are compact sets and the objective functions are continuous. Moreover, for the efficient sets of (MOP_1) and (MOP_2) external stability holds since the ordering cone $\mathbb{R}^m_+$ is a pointed closed convex cone and $h(X) := \{h(x) \mid x \in X\}$ is a compact set, compare with [27, Theorem 3.2.9]. This means that for any $x \in X$ there exists some $\tilde{x} \in X$ with $\tilde{x}$ efficient and $h(\tilde{x}) \leq h(x)$. As a consequence, the next theorem, which is given for instance in [18], also holds for our setting.

Theorem 1 *The set of minimal solutions of the constrained problem (P_1) always contains an efficient solution of the associated multiobjective counterpart problem (MOP_1), and all minimal solutions of (P_1) are weakly efficient for (MOP_1). Conversely, the set of efficient solutions of (MOP_1) contains at least one minimal solution of (P_1).*

Since the pair $((P_2), (MOP_2))$ is a special case of the pair $((P_1), (MOP_1))$, see Remark 1, Theorem 1 also holds for $((P_2), (MOP_2))$. As it was explained in [18], the optimal solution of (P_1) can be calculated as a specific efficient solution of (MOP_1):

Theorem 2 *Let X_E be the efficient set for (MOP_1) and $\bar{x} \in \mathrm{argmin}\{f(x) \mid x \in X_E \cap S\}$. Then $\bar{x}$ is minimal for (P_1).*

In general, the determination of the complete efficient set is not practicable if only the solution of a single-objective constrained optimization problem (P_1) is sought. However, if an algorithm can compute an approximation of the efficient set in a region of interest, it is possible to find near-optimal solutions of the constrained optimization problem in reasonable time, and additionally provide trade-off information between objective values and constraint satisfaction.

2.2 Lower bounding by convex underestimators

Branch-and-bound algorithms heavily rely on lower bounds for the values of scalar-valued functions on subsets. One possible approach to obtain lower bounds is the formulation of so

called convex underestimators. A *convex underestimator* of a function $f : \mathbb{R}^n \to \mathbb{R}$ on a box $X = [\underline{x}, \overline{x}] \subseteq \mathbb{R}^n$ is a convex function $\underline{f} : X \to \mathbb{R}$ with $\underline{f}(x) \le f(x)$ for all $x \in X$, see, for example, [1]. A convex underestimator for a twice continuously differentiable function f on X can, for example, be calculated as

$$\underline{f}(x) := f(x) + \frac{\alpha_f}{2}(\underline{x} - x)^T(\overline{x} - x),\tag{2}$$

where $\alpha_f \ge \max\{0, -\min_{x \in X} \lambda_{\min, f}(x)\}$. Here, $\lambda_{\min, f}(x)$ denotes the smallest eigenvalue of the Hessian $H_f(x)$ of f in x, see [20]. The minimal value of $\underline{f}$ over X, which can be calculated by standard techniques from convex optimization, delivers then a lower bound for the values of f on X. A lower bound for $\lambda_{\min, f}(x)$ over X can be calculated easily with the help of interval arithmetic, see again [20]. For that, the Matlab toolbox Intlab can efficiently be used [26]. See also [31] for improved lower bounds. The above proposed and some other methods to obtain lower bounds for $\lambda_{\min, f}(x)$ are described and tested in [1]. There are also other possibilities for the calculation of convex underestimators. For example in [1] special convex underestimators for bilinear, trilinear, fractional, fractional trilinear or univariate concave functions were defined. Here, we restrict ourselves to the above proposed convex underestimator. The theoretical results remain true in case that the above underestimators are replaced by tighter ones.

3 Relations between constrained optimization problems and their multiobjective counterparts

From Sect. 2 we already know that (P_1) and (P_2) are equivalent and that all optimal solutions of (P_1) and (P_2) are weakly efficient for (MOP_1) and (MOP_2), respectively.

Next to some obvious relations, we can state a relationship between the ε-minimal solutions of (P_1) and the ε-efficient solutions of (MOP_1). Due to Remark 1, an equivalent result holds for $((P_2),(MOP_2))$.

Lemma 1 *Let $\bar{x} \in S$ be ε-minimal for (P_1). Then $\bar{x}$ is ε-efficient for (MOP_1).*

Proof Since $\bar{x} \in S$ is ε-minimal for (P_1) it holds that $f(\bar{x}) < f(x) + \varepsilon$ for all $x \in S$. Now assume that $\bar{x}$ is not ε-efficient for (MOP_1). Then a solution $\tilde{x} \in X$ exists with

$$h(\tilde{x}) \leqq h(\bar{x}) - \varepsilon e,$$

i.e., $f(\tilde{x}) \le f(\bar{x}) - \varepsilon$ and $g_k(\tilde{x}) \le g_k(\bar{x}) - \varepsilon$ for all $k = 1, \ldots, r$ (with at least one strict inequality). Since $\bar{x} \in S$ it follows that $g_k(\tilde{x}) \le 0 - \varepsilon < 0$ for all $k = 1, \ldots, r$ and thus $\tilde{x}$ is feasible for (P_1). But with $f(\tilde{x}) \le f(\bar{x}) - \varepsilon$ we obtain a contradiction to the ε-minimality of $\bar{x}$ for (P_1). $\qquad\square$

The relationship between the different solution categories for (P_1) and (MOP_1) according to Theorem 1 and Lemma 1 are visualized in Fig. 1.

In contrast to the equivalence between (P_1) and (P_2), the optimization problems (MOP_1) and (MOP_2) are in general not equivalent if $r \ge 2$. However, we can state the following lemma.

Lemma 2 *Let $\bar{x} \in X$ be weakly efficient (or even efficient) for (MOP_2). Then $\bar{x} \in X$ is weakly efficient for (MOP_1).*
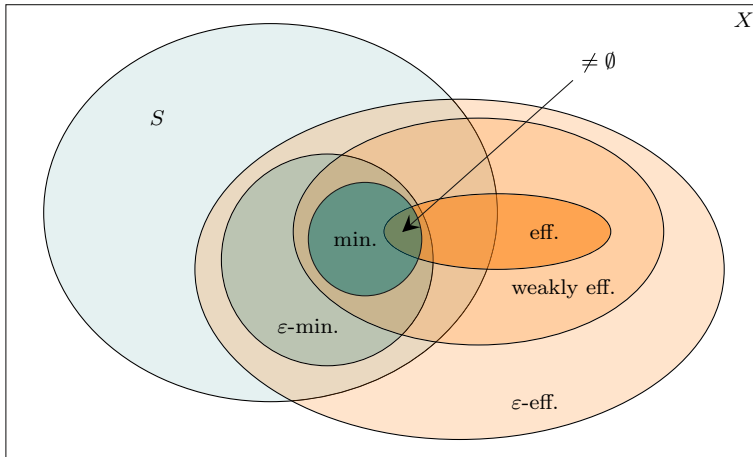
Fig. 1 Venn diagram showing the relations between the solutions of (P_1) and (MOP_1)

Proof Here we use $h := (f, g_1, \ldots, g_r)^T$. Assume that $\bar{x}$ is not weakly efficient for (MOP_1). Then there is an $x^* \in X$ with $h(x^*) < h(\bar{x})$, i.e., $f(x^*) < f(\bar{x})$ and $g_k(x^*) < g_k(\bar{x})$ for $k = 1, \ldots, r$. This implies $\max_{k=1,\ldots,r} g_k(x^*) < \max_{k=1,\ldots,r} g_k(\bar{x})$, which contradicts the assumption that $\bar{x}$ is weakly efficient (or efficient) for (MOP_2). $\qquad\square$

To see that the inverse implication does not hold, set $f, g_1, g_2 \colon \mathbb{R} \to \mathbb{R}$ with $f(x) = x^2$, $g_1(x) = x$, and $g_2(x) = -x$, and $X = [-5, 5]$. Then all $x \in X$ are weakly efficient for (MOP_1), but only $x = 0$ is weakly efficient for (MOP_2).

Note that for the special case that $r = 1$ the two problems (MOP_1) and (MOP_2) are equivalent by definition.

Lemma 3 *Let $X_E^{(MOP_1)}$ be the efficient set of (MOP_1) and $X_E^{(MOP_2)}$ be the efficient set of (MOP_2).*

(i) *Let $r = 1$ and $\bar{x} \in X_E^{(MOP_1)}$ with $\bar{x} \in \operatorname{argmax}\{g_1(x) \mid x \in S\}$. Then $\bar{x}$ is minimal for (P_1).*

(ii) *Let $r = 1$ and $\bar{x} \in X_E^{(MOP_1)}$ with $\bar{x} \in \operatorname{argmax}\{g_1(x) \mid x \in X_E^{(MOP_1)} \cap S\}$. Then $\bar{x}$ is minimal for (P_1).*

(iii) *Let $r \geq 2$ and $\bar{x} \in X_E^{(MOP_2)}$ with $\bar{x} \in \operatorname{argmax}\{\max_{k=1,\ldots,r} g_k(x) \mid x \in S\}$. Then $\bar{x}$ is minimal for (P_1) and (P_2).*

(iv) *Let $r \geq 2$ and $\bar{x} \in X_E^{(MOP_2)}$ with $\bar{x} \in \operatorname{argmax}\{\max_{k=1,\ldots,r} g_k(x) \mid x \in X_E^{(MOP_2)} \cap S\}$. Then $\bar{x}$ is minimal for (P_1) and (P_2).*

Proof (i) Assume that $\bar{x}$ is not minimal for (P_1). Then there is a point $x^* \in S$ with $f(x^*) < f(\bar{x})$ and $g_1(x^*) \leq 0$. Moreover, it holds $g_1(x^*) \leq g_1(\bar{x})$. But this contradicts the efficiency of $\bar{x}$ for (MOP_1).

(ii) Assume that $\bar{x}$ is not minimal for (P_1). Then there is a point $x^* \in S$ with $f(x^*) < f(\bar{x})$ and $g_1(x^*) \leq 0$. Since $\bar{x} \in X_E^{(MOP_1)}$ we obtain $g_1(x^*) > g_1(\bar{x})$. Therefore, $x^* \notin X_E^{(MOP_1)}$, since otherwise $\bar{x}$ would not be an element of $\operatorname{argmax}\{g_1(x) \mid x \in X_E^{(MOP_1)} \cap S\}$. For the optimization problem (MOP_1) external stability holds. Hence, since $x^* \notin X_E^{(MOP_1)}$, there exists a point $\tilde{x} \in X_E^{(MOP_1)}$ with $f(\tilde{x}) \leq f(x^*) < f(\bar{x})$ and

$g_1(\tilde{x}) \leq g_1(x^*) \leq 0$. The feasible solutions $\bar{x}$ and $\tilde{x}$ are both efficient of (MOP_1) and their images do not dominate each other. Therefore, $0 \geq g_1(\tilde{x}) > g_1(\bar{x})$ holds. This contradicts $\bar{x} \in \text{argmax}\{g_1(x) \mid x \in X_E^{(MOP_1)} \cap S\}$.

(iii) The assertion that $\bar{x}$ is minimal for (P_2) follows analogously to (i). Because of the equivalence of (P_1) and (P_2), the point $\bar{x}$ is also minimal for (P_1).

(iv) The assertion that $\bar{x}$ is minimal for (P_2) follows analogously to (ii). Because of the equivalence of (P_1) and (P_2), the point $\bar{x}$ is also minimal for (P_1).

$\square$

Note that a solution $\bar{x}$ according to Lemma 3, cases (i) and (iii), does not have to exist. Consider, for example, the simple linear problem $\min_{x \in S} f(x)$ with $f(x) = x$, $g_1(x) = x - 1$, $X = [0, 1]$, and $S = \{x \in X \mid x - 1 \leq 0\}$. Then $X_E^{(MOP_1)} = \{0\}$ while $\text{argmax}\{g_1(x) \mid x \in S\} = \{1\}$ (note that $x^* = 0$ is the unique minimal solution for (P_1) in this case). Moreover, Lemma 3, cases (iii) and (iv) with $\bar{x} \in X_E^{(MOP_1)}$ instead of $\bar{x} \in X_E^{(MOP_2)}$ do not hold in general. This is the reason why we focus on solving problem (MOP_2) rather than problem (MOP_1) in the following sections.
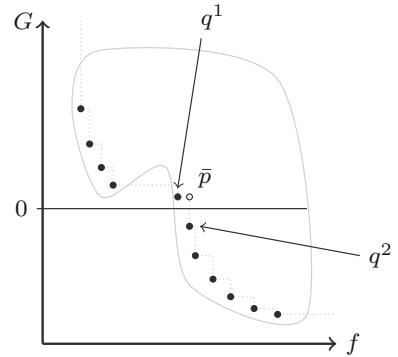
4 Multiobjective counterpart αBB algorithm

We will now focus on the constrained optimization problem (P_2) and its multiobjective counterpart (MOP_2) as suggested at the end of Sect. 3.

To solve (MOP_2) we adapt the branch-and-bound based algorithm for nonconvex multiobjective optimization which was given in [24]. It is based on an iterative subdivision of the feasible set $X = [\underline{x}, \overline{x}]$ into successively smaller boxes (see Sect. 4.1 below), combined with a bounding scheme that prunes irrelevant areas of the decision space. While this multiobjective αBB algorithm determines an approximation of the *complete* nondominated set, the *multiobjective counterpart αBB algorithm* (MOCPαBB algorithm) directs the search towards the region of interest for the constrained problem (P_2) in order to avoid unnecessary computations as much as possible. More precisely, given a required accuracy $\varepsilon > 0$, the multiobjective αBB algorithm from [24] terminates with an approximation of the complete nondominated set of (MOP_2) that consists of ε-nondominated points (i.e., images of ε-efficient solutions) such that for every nondominated point y there exists a representative $\bar{y}$ in the approximation such that $\bar{y} - \frac{\varepsilon}{2}e \leq y$. Recognizing the fact that in our case (MOP_2) is a multiobjective counterpart of a constrained optimization problem (P_2), we suggest appropriately adapted *discarding tests* (Sect. 4.2) as well as *selection* and *termination rules* (Sect. 4.3) to direct the search towards an optimal solution of the constrained problem. The overall MOCPαBB algorithm is stated in Sects. 4.4 and 4.5 verifies the existence of appropriate bounds.

4.1 Box generation and branching scheme

The subdivision of the feasible set $X = [\underline{x}, \overline{x}]$ of (MOP_2) in the decision space is implemented in the standard way of such subdivision algorithms. Thus, it is completely analogous to the subdivision as done in the multiobjective αBB algorithm of [24]: Starting from the initial box $X = [\underline{x}, \overline{x}]$, a series of subboxes is generated by iteratively splitting a currently selected box X^* perpendicularly to a longest edge into two subboxes X^1 and X^2. For the description of the selection rule for a box X^*, which determines in which series the boxes are subdivided, we refer to Sect. 4.3.

We denote the list of current, also called active, boxes by $\mathcal{L}_W$. The list $\mathcal{L}_W$ is also denoted as *working list* in the following, and it is initialized with X. The method terminates when the list $\mathcal{L}_W$ is empty. During the course of the algorithm, boxes that can not contribute to the approximation of the optimal set of problem (P_2) are discarded from further consideration. On the other hand, boxes that have achieved a certain accuracy w.r.t. the current bound sets and for which a further refinement is not promising are stored in a tentative *solution list* $\mathcal{L}_S$.

4.2 Bound computation and discarding test

A central element in the MOCPαBB algorithm is the discarding test which is used to prune selected boxes from the list $\mathcal{L}_W$ that do not contain (near) optimal solutions of (P_2). For this purpose, upper and lower bounds are computed based on solutions of (MOP_2) (upper bounds) and convex underestimators over selected subboxes (lower bounds), respectively.
Upper Bounds During the course of the MOCPαBB algorithm and while exploring selected boxes from $\mathcal{L}_W$, we generate a stable set $\mathcal{L}_{PNS}$ of feasible outcome vectors (called *provisional nondominated set*) representing upper bounds for the global nondominated points of (MOP_2). In this context, a set $\mathcal{N} \subset \mathbb{R}^m$ is called *stable* if for any $y^1, y^2 \in \mathcal{N}, y^1 \nleq y^2$ holds. Every time a point q is a new candidate for $\mathcal{L}_{PNS}$, we check if this point is dominated by any other point of $\mathcal{L}_{PNS}$. In this case, q will not be included in $\mathcal{L}_{PNS}$. Otherwise, q will be added to $\mathcal{L}_{PNS}$ and all points that are dominated by q are removed. Figure 2 shows an example for a provisional nondominated set $\mathcal{L}_{PNS}$ for an instance of (MOP_2). Note that in the biobjective case considered here, the points $y = (f, G)(x)$ in $\mathcal{L}_{PNS}$ can be ordered such that their objective function values $f(x)$ are strictly increasing while their largest constraint values $G(x) = \max_{k=1,...,r} g_k(x)$ are strictly decreasing.

Since we want to direct the search towards the region of interest of the constrained problem (P_2), we are particularly interested in the two points from $\mathcal{L}_{PNS}$ defined by

$$q^1 := \operatorname{argmin}_{p \in \mathcal{L}_{PNS}}\{p_2 \mid p_2 > 0\} \text{ and} \tag{3}$$

$$q^2 := \operatorname{argmax}_{p \in \mathcal{L}_{PNS}}\{p_2 \mid p_2 \leq 0\}, \tag{4}$$

see Fig. 2 for an illustration. Intuitively, these are the points of $\mathcal{L}_{PNS} \subseteq \mathbb{R}^2$ which are "around" $G(x) = 0$. Note that $q^1 = \operatorname{argmax}_{p \in \mathcal{L}_{PNS}}\{p_1 \mid p_2 > 0\}$ and $q^2 = \operatorname{argmin}_{p \in \mathcal{L}_{PNS}}\{p_1 \mid p_2 \leq 0\}$ also hold. It is possible that one of the two points does not exist. This case is considered in Sect. 4.5. For the remainder of this section, we assume that both points q^1 and q^2 exist. Note that, as the list $\mathcal{L}_{PNS}$ changes its cardinality and precise-

ness during the algorithm, the points q^1 and q^2 generally change their position during the procedure.

Lower Bounds A main ingredient of the adapted discarding test in our new MOCPαBB algorithm is to calculate lower bounds for the image set of a selected box X^* from $\mathcal{L}_W$, i.e., a set LB such that $LB + \mathbb{R}_+^2$ is a superset of $\{y \in \mathbb{R}^2 \mid \exists x \in X^* : y_1 = f(x), y_2 = G(x)\}$, and compare them with the upper bounds q^1 and q^2 of the interesting part of the nondominated set.

As a lower bound for the image set of a box X^*, we use the so called *ideal point* of convex underestimators of the objective functions on the considered box X^*. In general, the ideal point of a vector-valued function consists of the global minimal values of each function. For (MOP_2) this is the point $a = (a_1, a_2)^T$ with

$$a_1 = \min_{x \in X^*} f(x) \text{ and } a_2 = \min_{x \in X^*} G(x) = \min_{x \in X^*} (\max_{k=1,\dots,r} g_k(x)).$$

Determining a in the nonconvex case is not possible without applying techniques from global optimization, which are time consuming. As we have to calculate ideal points repeatedly over many subboxes, we use convex underestimators as defined in Sect. 2.2 for both objective functions. The ideal point of the convex underestimators is denoted by $(\underline{f}, \underline{G})^T$ and is calculated as

$$\underline{f} = \min_{x \in X^*} f_\alpha(x) \text{ and } \underline{G} = \min_{x \in X^*} G_\beta(x), \tag{5}$$

where f_α is a convex underestimator of f on X^* and G_β is a convex underestimator of G on X^*. While we can directly use the convex underestimator (2) to define f_α, the particular structure of G as the maximum of r functions suggests several alternative definitions based on (2). In the following lemma, we propose two such definitions for G_β.

Lemma 4 *Let a box $X^* = [\underline{x}^*, \overline{x}^*]$ be given. Moreover, let $g_k \colon \mathbb{R}^n \to \mathbb{R}$, $k = 1, \dots, r$ be twice continuously differentiable functions and let $g_{k,\beta_k} \colon \mathbb{R}^n \to \mathbb{R}$ be the corresponding convex underestimators on X^* according to (2), i.e.,*

$$g_{k,\beta_k}(x) := g_k(x) + \frac{\beta_k}{2}(\underline{x}^* - x)^T(\overline{x}^* - x)$$

with sufficiently large $\beta_k > 0$. Define the map $G \colon \mathbb{R}^n \to \mathbb{R}$ by $G(x) := \max_{k=1,\dots,r} g_k(x)$. Then the functions $G_\beta, G_{\tilde{\beta}} \colon \mathbb{R}^n \to \mathbb{R}$ with

$$G_\beta(x) = \max_{k=1,\dots,r} g_{k,\beta_k}(x)$$

$$G_{\tilde{\beta}}(x) = \left(\max_{k=1,\dots,r} g_k(x)\right) + \frac{\tilde{\beta}}{2}(\underline{x}^* - x)^T(\overline{x}^* - x) \text{ with } \tilde{\beta} := \max_{k=1,\dots,r} \beta_k$$

are convex underestimators of G on X^*.

Proof Follows immediately from [20] and the fact that the maximum of convex functions is again a convex function. □

While $G_{\tilde{\beta}}$ is probably the more intuitive choice for a convex underestimator since it follows directly from (2), the convex underestimator G_β is generally preferable since it provides stronger lower bounds. To illustrate this, consider an optimization problem with two constraints g_1 and g_2 where $g_1(x) \geq g_2(x)$ for all $x \in X^*$. Additionally, assume that g_1 is a convex function and g_2 is nonconvex. Therefore, $\beta_1 = 0$ and $\beta_2 > 0$, and hence

$g_{2,\beta_2}(x) \leq g_2(x) \leq g_1(x) = g_{1,\beta_1}(x)$ hold for all $x \in X^*$. Then, we obtain for the convex underestimators for G on X^*:

$$G_\beta(x) = g_1(x) \text{ and } G_{\tilde{\beta}}(x) = g_1(x) + \frac{\beta_2}{2}(\underline{x}^* - x)^T(\overline{x}^* - x),$$

and hence $G_\beta(x) \geq G_{\tilde{\beta}}(x)$ holds for all $x \in X^*$. The following lemma shows that this is true in general, i.e., G_β is indeed a stronger convex underestimator than $G_{\tilde{\beta}}$.

Lemma 5 *Under the assumptions of Lemma 4, it holds that $G_{\tilde{\beta}}(x) \leq G_\beta(x)$ for all $x \in X^*$.*

Proof Let $x \in X^*$ be arbitrary but fixed, and let $k^* \in \{1, \ldots, r\}$ be an index for which $g_{k^*}(x) = \max_{k=1,\ldots,r} g_k(x)$. Then $\tilde{\beta} = \max_{k=1,\ldots,r} \beta_k \geq \beta_{k^*}$ and hence

$$
\begin{aligned}
G_{\tilde{\beta}}(x) &= \left(\max_{k=1,\ldots,r} g_k(x) \right) + \frac{\tilde{\beta}}{2} \underbrace{(\underline{x}^* - x)^T}_{\leq 0} \underbrace{(\overline{x}^* - x)}_{\geq 0} \\
&\leq g_{k^*,\beta_{k^*}}(x) + \frac{\beta_{k^*}}{2}(\underline{x}^* - x)^T(\overline{x}^* - x) \\
&\leq \max_{k=1,\ldots,r} g_{k,\beta_k}(x) = G_\beta(x).
\end{aligned}
$$

$\square$

Thereby, G_β is the preferred convex underestimator and is used in our procedure. As G_β is in general not differentiable, we solve the smooth convex optimization problem to obtain the minimum of G_β on the box X^*:

$$\min_{x \in X^*, \, t \in \mathbb{R}} t \text{ s.t. } g_{k,\beta_k}(x) \leq t, \quad k = 1, \ldots, r. \tag{6}$$

Discarding Tests The following lemma is the basis for the first two discarding tests.

Lemma 6 *Let q^1 and q^2 be defined as in (3) and (4). Moreover, let $(\underline{f}, \underline{G})^T$ be a lower bound vector, where $\underline{f}$ and $\underline{G}$ are lower bounds for f and G on X^*, respectively. If $\underline{f} > q_1^2$ or $\underline{G} > q_2^1$ holds, the box X^* does not contain any minimal solution of (P_2).*

Proof If the first condition $\underline{f} > q_1^2$ holds, we obtain $q_1^2 < \underline{f} \leq f(x)$ for all $x \in X^*$. Moreover, the point q^2 is the objective vector of a solution $\tilde{x} \in X$, and by (4) it holds $G(\tilde{x}) = q_2^2 \leq 0$. Hence, q^2 is an image of a feasible solution $\tilde{x}$ of (P_2) and the objective function value $q_1^2 = f(\tilde{x})$ is less than any function value attainable in X^*. Thus, there is no minimal solution for (P_2) in X^* in this case.

If $\underline{G} > q_2^1$ holds, by (3) it follows $0 < q_2^1 < \underline{G} \leq G(x) = \max_{k=1,\ldots,r} g_k(x)$ for all $x \in X^*$. Hence, for every $x \in X^*$ there is at least one constraint g_k with $g_k(x) > 0$. This means that every $x \in X^*$ is infeasible for (P_1) and cannot be a minimal solution of (P_1). $\square$

Note that the set $\{(\underline{f}, \underline{G})^T\} + \mathbb{R}_+^2$ can be interpreted as an *outer approximation* of the convex set

$$\mathcal{P} := \{y \in \mathbb{R}^2 \mid y_1 = f_\alpha(x), \, y_2 = G_\beta(x), \, x \in X^*\} + \mathbb{R}_+^2 \tag{7}$$

and that the discarding test of Lemma 6 actually tests whether $\bar{p} := (q_1^2, q_2^1)^T \notin \{(\underline{f}, \underline{G})^T\} + \mathbb{R}_+^2$ (and hence $\bar{p} \notin \mathcal{P}$), see Fig. 2 for an illustration of $\bar{p}$. In this context, an outer approximation of a set $A \subseteq \mathbb{R}^2$ is given by a set $LB \subseteq \mathbb{R}^2$ such that $A \subseteq LB + \mathbb{R}_+^2$.

While the discarding test of Lemma 6 is very simple and can be easily evaluated, it is rather weak in practice. In order to obtain a tighter outer approximation of the set $(f_\alpha, G_\beta)(X^*)$

and, hence, to strengthen the discarding test, we replace the ideal point of the convex under-estimators by an improved lower bound set LB that is implicitly obtained from solving a single-objective convex optimization problem to decide whether $\bar{p}$ belongs to $\mathcal{P}$:

$$\min_{(x,t)\in\mathbb{R}^{n+1}} t \text{ s.t. } x \in X^*,$$
$$\bar{p}_1 + t \geq f_\alpha(x), \qquad\qquad (P^*_{X^*,\bar{p}})$$
$$\bar{p}_2 + t \geq g_{k,\beta_k}(x), \quad k = 1, \ldots, r.$$

A minimal solution of $(P^*_{X^*,\bar{p}})$ is named $(\tilde{x}, \tilde{t})$. If $\tilde{t} \leq 0$ holds, we have $\bar{p} \in \mathcal{P}$. Otherwise, if $\tilde{t} > 0$ holds, the point $\bar{p}$ lies outside $\mathcal{P}$ and can thus be separated from $\mathcal{P}$ with a supporting hyperplane that can be constructed from a Lagrange multiplier $\lambda^* \in \mathbb{R}^{1+r}$ for the inequality constraints of $(P^*_{X^*,\bar{p}})$: Given λ^*, a normal vector of this hyperplane is obtained as $(\lambda_1^*, 1 - \lambda_1^*) = (\lambda_1^*, \sum_{k=2}^{r+1} \lambda_k^*) \in \mathbb{R}^2$. Moreover, a support vector of this hyperplane is $\bar{p} + \tilde{t}e$. This result can be shown by using necessary and sufficient optimality conditions for nonsmooth, convex optimization problems with linear constraints.

Note that the outer approximation of $\mathcal{P}$ gets indeed improved by using the constructed supporting hyperplane, since the point $\bar{p}$ is infeasible for this improved approximation.

The next lemma formally proves the above mentioned fact that a box can be discarded for $\tilde{t} > 0$.

Lemma 7 *Let $\mathcal{L}_{PNS}$ be a stable subset of $h(X)$, $X^* \subseteq \mathbb{R}^n$ be a box, f_α and G_β be the convex underestimators of f and G on X^* and q^1, q^2 be defined as in (3) and (4). Moreover, consider the optimization problem $(P^*_{X^*,\bar{p}})$ with $\bar{p} = (q_1^2, q_2^1)^T$ with the minimal solution $(\tilde{x}, \tilde{t})$. If $\tilde{t} > 0$ holds, then X^* does not contain any minimal solution for (P_1).*

Proof Assume that there is a minimal solution x^* for (P_1) with $x^* \in X^* \cap S$. The point q^2 is the function value of a feasible solution $\tilde{x} \in S$, i.e., there is an $\tilde{x} \in S$ with

$$q_1^2 = f(\tilde{x}). \qquad\qquad (8)$$

Consider the pair $(x^*, 0) \in \mathbb{R}^{n+1}$. Using the minimality and feasibility of x^* and properties of convex underestimators, we obtain that this pair is feasible for $(P^*_{X^*,\bar{p}})$:

$$\bar{p}_1 + 0 \geq f_\alpha(x^*), \quad \text{because } \bar{p}_1 = q_1^2 = f(\tilde{x}) \geq f(x^*) \geq f_\alpha(x^*)$$
$$\bar{p}_2 + 0 \geq g_{k,\beta_k}(x^*), \quad \text{because } \bar{p}_2 = q_2^1 > 0 \geq g_k(x^*) \geq g_{k,\beta_k}(x^*)$$
$$\text{for all } k = 1, \ldots, r.$$

This is a contradiction to the minimality of $\tilde{t} > 0$. □

Note that the optimization problem $(P^*_{X^*,\bar{p}})$ is also used for a termination procedure which is discussed in Sect. 4.3 below.

4.3 The selection and termination rule

A selection rule determines within the branch-and-bound procedure which of the remaining subboxes is selected next to be bisected. The aim of every selection rule is to identify boxes whose subboxes deliver good bounds for the globally minimal value. Such rules are usually heuristic. A common rule in multiobjective optimization is to choose the box which has the smallest lower bound w.r.t. one (arbitrary but fixed) objective function, or w.r.t. a weighted sum of all objectives. In our case, we are interested in the minimization of f, but also in

finding at least one feasible solution of (P_1). Therefore, as long as no point q^2 exists, a box with the smallest lower bound for G is selected as the next box which will be bisected, while a box with the smallest calculated lower bound for f is chosen otherwise.

The termination rule decides whether a box is discarded, whether it is stored in the solution list $\mathcal{L}_S$ (and thus not further bisected), or whether it is stored in the working list $\mathcal{L}_W$ (i.e., the list of active boxes that will be analyzed further). In the work of [24], a termination procedure is introduced which guarantees some accuracies of calculated points in the decision and in the objective space. The minimal solution of the optimization problem ($P^*_{X^*, \bar{p}}$) plays an essential role for the decision to store a box in the solution list. For our algorithm we use a simpler rule in the first part, which ensures an accuracy of q^1 and q^2:

1. Termination rule: *Solve* ($P^*_{X^*, \bar{p}}$) *for the box X^* and $\bar{p} = (q_1^2, q_2^1)^T$, where q^1 and q^2 are defined as in* (3) *and* (4). *The minimal solution is denoted by $(\tilde{x}, \tilde{t})$. Store X^* in $\mathcal{L}_S$ if $0 \geq \tilde{t} \geq -\frac{\varepsilon}{2}$ holds for a given $\varepsilon > 0$.*

Note that in case of $\tilde{t} > 0$, the box is discarded because of Lemma 7.

4.4 The MOCPαBB algorithm

As mentioned in Sect. 4.1, the MOCPαBB algorithm is based on a box subdivision of $X = [\underline{x}, \overline{x}]$. For all not-discarded boxes X^* in the working list $\mathcal{L}_W$ and in the solution list $\mathcal{L}_S$, an outer approximation of the set $\mathcal{P}$ (which is an underestimation of possible outcome vectors on X^*, see (7)) is encoded in a set $\mathcal{H}$. This outer approximation is initialized using the ideal point $(\underline{f}, \underline{G})$ of $\mathcal{P}$ and further refined whenever additional supporting hyperplanes of $\mathcal{P}$ become available. Every element of $\mathcal{H}$ is a pair consisting of the normal vector λ of the hyperplane and a scalar value, which is the scalar product of λ and one point belonging to the hyperplane. Thus, the initial outer approximation of $\mathcal{P}$ obtained by minimizing the convex underestimators on the box X^* is given by

$$\mathcal{H} = \{((1, 0)^T, \underline{f}), ((0, 1)^T, \underline{G})\}.$$

Hence, the lower bounds $\underline{f}$ and $\underline{G}$ for each objective are stored in $\mathcal{H}$ as well.

The complete algorithm consists of two parts. The first part stated above is the main part and includes the discarding test as well as the selection and termination rule which are explained in the previous sections. The second part is a post processing algorithm which includes different (optional) steps in order to further improve the solution quality.

The following theorem states that the algorithm (without the post processing) terminates. We omit the proof as it is a special case of the proof of Lemma 4.1 of [24].

Theorem 3 *Algorithm* 1 *needs finitely many subdivisions in line* 7.

Next, we want to derive some properties of the points q^1 and q^2 computed with Algorithm 1. These properties will be used in a post processing for further improvements.

Theorem 4 *Let q^1, q^2 be defined by* (3) *and* (4) *after Algorithm* 1, *line* 26. *In particular, assume that both exist. Denote the pre-images of q^1 and q^2 by x^1 and x^2, respectively. Then x^1 and x^2 are ε-efficient of* (MOP_1) *and* (MOP_2). *Additionally, to each x^1 and x^2 exist some boxes $X^1, X^2 \in \mathcal{L}_S$ with $x^1 \in X^1$ and $x^2 \in X^2$.*

Proof The solution x^1 is the pre-image of q^1, i.e., it holds $f(x^1) = q_1^1$ and $\max_{k=1,\dots,r} g_k(x^1) = G(x^1) = q_2^1$. Assume that x^1 is not ε-efficient for (MOP_1) or for (MOP_2). Then there is a solution $x' \in X$ with

$$f(x') \leq q_1^1 - \varepsilon \tag{9}$$

Algorithm 1 MOCPαBB algorithm to solve (P_2)

INPUT: (P_2), $\varepsilon > 0$, $\delta > 0$
OUTPUT: $\mathcal{L}_S$, q^1, q^2, $(\mathcal{A}, x_{\min}, UB)$
1: $\mathcal{H}^0 \leftarrow \{((1,0)^T, -\infty), ((0,1)^T, -\infty)\}$, $\mathcal{L}_W \leftarrow \{(X, \mathcal{H}^0)\}$, $\mathcal{L}_S \leftarrow \emptyset$, $\mathcal{L}_{PNS} = \emptyset$, $\mathcal{A} = \emptyset$
2: **while** $\mathcal{L}_W \neq \emptyset$ **do**
3:     **if** a point q^2 exists **then**
4:         Select a pair $(X^*, \mathcal{H})$ with a smallest value $\underline{f}$ from $\mathcal{L}_W$, delete it from $\mathcal{L}_W$
5:     **else** Select a pair $(X^*, \mathcal{H})$ with a smallest value $\underline{G}$ from $\mathcal{L}_W$, delete it from $\mathcal{L}_W$
6:     **end if**
7:     Bisect X^* perpendicularly to a direction of maximum width $\rightarrow X^1, X^2$
8:     **for** $l = 1, 2$ **do**
9:         Compute for f and $G := \max_{k=1,\dots,r} g_k(\cdot)$ their convex underestimators on X^l and their
        corresponding minimal solution x^f and x^G
10:         $\mathcal{H} \leftarrow \{((1,0)^T, f_\alpha(x^f)), ((0,1)^T, G_\beta(x^G))\}$
11:         Store image of x^f and x^G in $\mathcal{L}_{PNS}$ and update $\mathcal{L}_{PNS}$ to a stable set
12:         Obtain q^1, q^2 and $\bar{p}$ from $\mathcal{L}_{PNS}$ by (3), (4) and (11)
13:         **if** $f_\alpha(x^f) > q_1^2$ or $G_\beta(x^G) > q_2^1$ **then**
14:             Discard X^l
15:         **else**
16:             Solve $(P^*_{X^*, \bar{p}})$ with minimal solution $(\tilde{x}, \tilde{t})$ and Lagrange multiplier $\bar{\lambda}$, and set $\lambda \leftarrow$
            $(\bar{\lambda}_1, 1 - \bar{\lambda}_1)$
17:             **if** $\tilde{t} > 0$ **then**
18:                 Discard X^l
19:             **else if** $\tilde{t} \geq -\frac{\varepsilon}{2}$ **then**
20:                 $\mathcal{H} \leftarrow \mathcal{H} \cup \{(\lambda, \lambda^T(\bar{p} + \tilde{t}e))\}$
21:                 Store $(X^l, \mathcal{H})$ in $\mathcal{L}_S$
22:             **else** Store $(X^l, \mathcal{H})$ in $\mathcal{L}_W$
23:             **end if**
24:         **end if**
25:     **end for**
26: **end while**
27: $(\mathcal{L}_{S_3}, \mathcal{A}, x_{\min}, UB) \leftarrow$**Post processing**$(\mathcal{L}_S, q^1, q^2, \bar{p}, \delta)$

$$g_k(x') \leq G(x') \leq q_2^1 - \varepsilon \text{ for all } k = 1, \dots, r, \qquad (10)$$

and at least one inequality is strict. First assume that x' belongs to a box $X' \in \mathcal{L}_S$. Then the termination rule implies that the minimal solution of $(P^*_{X^*, \bar{p}})$ is $(\tilde{x}, \tilde{t})$ with $-\frac{\varepsilon}{2} \leq \tilde{t} \leq 0$. Consider the pair $(x', -\varepsilon) \in \mathbb{R}^{n+1}$ for the optimization problem $(P^*_{X^*, \bar{p}})$ with $\bar{p} = (q_1^2, q_2^1)$ as before. Then, with (9) and (10) we obtain

$$x' \in X',$$
$$q_1^2 + (-\varepsilon) \geq q_1^1 - \varepsilon \geq f(x') \geq f_\alpha(x'),$$
$$q_2^1 + (-\varepsilon) \geq G(x') \geq g_k(x') \geq g_{k,\beta_k}(x') \text{ for all } k = 1, \dots, r$$

All constraints of $(P^*_{X^*, \bar{p}})$ are satisfied. Thus, $(x', -\varepsilon)$ is feasible for $(P^*_{X^*, \bar{p}})$, but this contradicts the minimality of $\tilde{t} \geq -\frac{\varepsilon}{2}$.

We can conclude that, since the algorithm terminates, x' has to belong to a box X' which was discarded. There are two conditions based on which a box can be discarded. Either because of the test from Lemma 7 or from Lemma 6. As $(x', -\varepsilon)$ is still feasible for $(P^*_{X^*, \bar{p}})$, we can see that X' was not discarded because of Lemma 7.

Let $(\underline{f}', \underline{G}')$ be the ideal point of the convex underestimators of f and G on X'. Then the two reasons for discarding X' from Lemma 6 are still possible: (a) $\underline{f}' > q_1^2$ and (b) $\underline{G}' > q_2^1$.

Fig. 3 Example where pre-image of q^2 is not ε-minimal of (P_1)
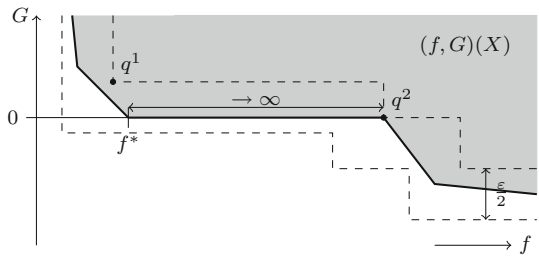
For case (a), we obtain with (9) the following chain of inequalities:

$$q_1^1 < q_1^2 < \underline{f}' \leq f_\alpha(x') \leq f(x') \leq q_1^1 - \varepsilon.$$

This is a contradiction to $\varepsilon > 0$. Case (b) leads with (10) to

$$q_2^1 < \underline{G}' \leq G_\beta(x') \leq G(x') \leq q_2^1 - \varepsilon,$$

which contradicts $\varepsilon > 0$ as well. Hence, x' does not exist and x^1 is ε-efficient for (MOP_1) and (MOP_2).

Now choose any box X^*, which is considered during Algorithm 1, with $x^1 \in X^*$. Let $(\underline{f}, \underline{G})$ be the ideal point of convex underestimators of f and G on X^*. As

$$\underline{f} = \min_{x \in X^*} f_\alpha(x) \leq f_\alpha(x^1) \leq f(x^1) = q_1^1 < q_1^2 \text{ and}$$
$$\underline{G} = \min_{x \in X^*} G_\beta(x) \leq G_\beta(x^1) \leq G(x^1) = q_2^1$$

hold even for the final points q^1, q^2, X^* can not have been discarded by Lemma 6. In addition, with $(x^1, 0) \in \mathbb{R}^{n+1}$ a feasible solution for ($P^*_{X^*, \bar{p}}$) is found and thus, for the minimal value $\tilde{t}$ of this optimization problem $\tilde{t} \leq 0$ holds. Hence, X^* does not get discarded. As the algorithm terminates (as shown in Theorem 3) there exists a subbox X^1 with $x^1 \in X^1 \subseteq X^*$ which is stored in $\mathcal{L}_S$.

The proof for x^2 is analogous. Note that $G(x^2) \leq 0 < q_2^1$ holds.   □

Figure 3 shows that the pre-image of q^2 does not have to be ε-minimal for the single-objective problem (P_1). Indeed, even in cases where $q_2^2 = 0$ holds, $f(x^2) = q_1^2$ may be arbitrarily far away from the optimal value f^*.

Nevertheless, the following lemma states that we can enclose the image point of a minimal solution for (P_1) by a tube-shaped set determined by $\bar{p}$ and ensure the ε-minimality for (P_1) of the pre-image of q^2 in a special case.

Lemma 8 *Let x^* be a minimal solution of (P_1). Then*

$$(f, G)(x^*) \in (\{\bar{p}\} - \mathbb{R}_+^2) \setminus (\{\bar{p} - \frac{\varepsilon}{2}e\} - \text{int}(\mathbb{R}_+^2)).$$

A direct consequence from this is that if $q_2^1 > \frac{\varepsilon}{2}$ holds, the pre-image of q^2 is ε-minimal for (P_1) (actually ($\frac{\varepsilon}{2} + \mu$)-minimal for all $\mu > 0$).

Proof Since x^* is minimal for (P_1) and because of the definition of q^1 and q^2 (see (3) and (4)), we know that $f(x^*) \leq q_1^2$ and $G(x^*) \leq 0 < q_2^1$. Hence, $(f, G)(x^*) \in \{(q_1^2, q_2^1)^T\} - \mathbb{R}_+^2 =$

$\{\bar{p}\} - \mathbb{R}_+^2$ holds. Assume now that $(f, G)(x^*) \in \{\bar{p} - \frac{\varepsilon}{2}e\} - \operatorname{int}(\mathbb{R}_+^2)$, i.e., $f(x^*) < q_1^2 - \frac{\varepsilon}{2}$ and $g(x^*) < q_2^1 - \frac{\varepsilon}{2}$. Then we can choose a parameter $\mu' > 0$ such that

$$f(x^*) \leq q_1^2 - (\frac{\varepsilon}{2} + \mu') \text{ and } g(x^*) \leq q_2^1 - (\frac{\varepsilon}{2} + \mu').$$

Let X^* be a box from the solution list $\mathcal{L}_S$ with $x^* \in X^*$. Because of Lemma 7 and the termination rule, X^* was not discarded, i.e., for the minimal solution $(\tilde{x}, \tilde{t})$ of $(P_{X^*, \bar{p}}^*)$ it holds $0 \geq \tilde{t} \geq -\frac{\varepsilon}{2}$. But $(x^*, -(\frac{\varepsilon}{2} + \mu))$ is feasible for $(P_{X^*, \bar{p}}^*)$, which contradicts the minimality of $\tilde{t} \geq -\frac{\varepsilon}{2}$. This shows the first part of the result, i.e., that $(f, G)(x^*) \in (\{\bar{p}\} - \mathbb{R}_+^2) \setminus (\{\bar{p} - \frac{\varepsilon}{2}e\} - \operatorname{int}(\mathbb{R}_+^2))$ holds.

The second part follows immediately: As $(f, G)(x^*) \notin \{\bar{p} - \frac{\varepsilon}{2}e\} - \operatorname{int}(\mathbb{R}_+^2)$, we have two possible cases: $G(x^*) \geq q_2^1 - \frac{\varepsilon}{2}$ or $f(x^*) \geq q_1^2 - \frac{\varepsilon}{2}$. Because of $q_2^1 > \frac{\varepsilon}{2}$, we obtain $q_2^1 - \frac{\varepsilon}{2} > 0$, and hence $G(x^*) \geq q_2^1 - \frac{\varepsilon}{2}$ can not hold because of the feasibility of x^* for (P_1). Hence, $f(x^*) \geq q_1^2 - \frac{\varepsilon}{2}$ holds. Using the minimality of x^*, it follows that

$$q_1^2 \leq f(x^*) + \frac{\varepsilon}{2} < f(x^*) + \frac{\varepsilon}{2} + \mu \leq f(x) + \frac{\varepsilon}{2} + \mu \text{ for all } \mu > 0, \ x \in S.$$

Consequently, the pre-image of q^2 is $(\frac{\varepsilon}{2} + \mu)$-minimal for (P_1), and, in particular, ε-minimal for (P_1). $\qquad\square$

4.5 Existence of q^1 and q^2

Recall the definitions of q^1 and q^2 from (3) and (4). The list $\mathcal{L}_{PNS}$ is not empty after a first element is added to this list at the beginning. Therefore, at least one of the two points q^1, q^2 exists. However, it is possible that only one of the two points exists during the whole course of the algorithm. This is, for example, the case when $X = S$ or $S = \emptyset$ holds. Nevertheless, we can still define a suitable point $\bar{p}$ which can be used for initializing the discarding test from Lemma 7.

$$\bar{p} := \begin{cases} (q_1^2, q_2^1)^T, & \text{if } q^1 \text{ and } q^2 \text{ exist} \\ (q_1^2, M)^T, & \text{if only } q^2 \text{ exists} \\ (M, q_2^1)^T, & \text{if only } q^1 \text{ exists} \end{cases} \tag{11}$$

Thereby, the constant real number M has to be chosen as an upper bound of $f(\cdot) + \varepsilon$ and $G(\cdot) + \varepsilon$ on the box X, i.e.,

$$M \geq \max\{\max_{x \in X} f(x), \max_{x \in X} G(x)\} + \varepsilon.$$

Such an upper bound can be calculated by interval arithmetic, for instance. If one of the two points q^1, q^2 does not exist, it is clearly not possible to apply the corresponding discarding test from Lemma 6 which depends on the missing point. Recall from Sect. 4.3 that the rule to select a box from $\mathcal{L}_W$ depends on the existence of q^2 as well. As long as q^2 does not exist, a box with a smallest lower bound for G is chosen, otherwise the box with a smallest lower bound for f is selected.

By the next lemma, we observe that in case that feasible solutions for problem (P_2) exist, the algorithm is able to find at least a so-called $\frac{\varepsilon}{2}$-*feasible* solution for (P_2), i.e., it finds a solution $x \in \mathbb{R}^n$ with $G(x) \leq \frac{\varepsilon}{2}$.

Lemma 9 *If $S \neq \emptyset$ holds, Algorithm 1 finds at least an $\frac{\varepsilon}{2}$-feasible solution for (P_2) (within the main* while*-loop).*

Proof If Algorithm 1 finds a solution x with $(f, G)(x) = q^2$, then x is feasible for (P_2) and, thus, also $\frac{\varepsilon}{2}$-feasible. We thus have to consider the case that q^2 does not exist. Then q^1 exists and $\bar{p} = (M, q_2^1)^T$. If $q_2^1 = \bar{p}_2 \leq \frac{\varepsilon}{2}$, then q^1 is $\frac{\varepsilon}{2}$-feasible and the statement is satisfied. Otherwise, we have that $q_2^1 = \bar{p}_2 > \frac{\varepsilon}{2}$.

Every time a box X^* that contains feasible solutions for (P_2) is considered, we obtain for this box a lower bound $\underline{G} \leq 0$ for G. Define $x^f \in \text{argmin}\{f_\alpha(x) \mid x \in X^*\}$ and $x^G \in \text{argmin}\{G_\beta(x) \mid x \in X^*\}$. Then the images of x^f and x^G are possible candidates for $\mathcal{L}_{PNS}$. If one of x^f and x^G is feasible for (P_2), then the algorithm has found a point q^2. Similarly, if one of x^f and x^G is $\frac{\varepsilon}{2}$-feasible for (P_2), then the image of this point is added to $\mathcal{L}_{PNS}$ and an $\frac{\varepsilon}{2}$-feasible solution is found. Otherwise, $G(x^f) > \frac{\varepsilon}{2}$ and $G(x^G) > \frac{\varepsilon}{2}$ hold, i.e., x^f and x^G are both infeasible for (P_2). In this case, problem ($P^*_{X^*, \bar{p}}$) is equivalent to

$$\min_{(x,t)\in\mathbb{R}^{n+1}} t \text{ s.t. } x \in X^*,$$
$$M + t \geq f_\alpha(x),$$
$$\bar{p}_2 + t \geq G_\beta(x).$$

Note that the first constraint induced by $\bar{p}$, $M + t \geq f_\alpha(x)$, is satisfied for all $x \in X^*$ and for all $t \geq -\varepsilon$ if M is chosen as described above. Moreover, $\bar{p}_2 > \frac{\varepsilon}{2}$ and $G_\beta(x^G) \leq 0$ (recall that $x^G \in X^*$) imply that an optimal solution $(\tilde{x}, \tilde{t})$ of problem ($P^*_{X^*, \bar{p}}$) satisfies $\tilde{t} < -\frac{\varepsilon}{2}$. Thus, as long as no $\frac{\varepsilon}{2}$-feasible solution is found, no box with $\underline{G} \leq 0$ is discarded.

Assume that this is the case every time a box X^* with feasible solutions is considered. Recall that the convex underestimators get better for smaller box widths, see [1]. Thus, if a certain box width is reached, we get $G(x) - G_\beta(x) \leq \frac{\varepsilon}{2}$ for all $x \in X^*$ and thus also for $x^G \in X^*$. Moreover, $\underline{G} = G_\beta(x^G) \leq 0$ and thus

$$G(x^G) - \frac{\varepsilon}{2} \leq G_\beta(x^G) \leq 0. \tag{12}$$

Now either $G(x^G) \leq \frac{\varepsilon}{2}$, and x^G is $\frac{\varepsilon}{2}$-feasible, or $G(x^G) > \frac{\varepsilon}{2}$, but then with (12) we obtain a contradiction.   □

5 Post processing: filtering, box refinement, and local search

The post processing further improves the so far found $\frac{\varepsilon}{2}$-efficient solutions. As before, we assume that the algorithm has found until now two outcome vectors q^1 and q^2. The pre-image of q^2 is a feasible solution for (P_2) and thus q_1^2 is an upper bound for the minimal value f^* of (P_2). In the special case that q^2 does not exist, the algorithm continues with $\bar{p} = (M, q_2^1)$ as described in Sect. 4.5. The pre-image of q^1 is infeasible for (P_2). If it is just slightly infeasible, i.e., if $q_2^1 \leq \frac{\varepsilon}{2}$, the two procedures **Box Refinement** and **Local Search and Adaptive Discarding** aim to find better upper bounds for f^* than q_1^2. Otherwise, i.e., in case in which q^1 is farer above 0, we can prove that the pre-image of q^2 is already ε-minimal for (P_2), see Lemma 8. We summarize the proposed steps in Algorithm 2.

5.1 Filtering

The first post processing step eliminates further boxes from the solution list. The reason for that is that boxes can be stored in the solution list during Algorithm 1 at an early iteration where the bounds q^1 and q^2 may be quite weak, but the termination rule is satisfied. As the

---

Algorithm 2 Post processing

INPUT: (P_2), $\mathcal{L}_S$, q^1, q^2, $\bar{p}$, δ
OUTPUT: $\mathcal{L}_{S_2}$ or $(\mathcal{L}_{S_3}, \mathcal{A}, x_{\min}, UB)$
1: $\mathcal{L}_{S_2} \leftarrow$ **Filtering**$(\mathcal{L}_S, q^1, q^2, \bar{p})$
2: **if** $q_2^1 \leq \frac{\varepsilon}{2}$ **then**
3:    $(\mathcal{L}_{S_3}, \mathcal{A}) \leftarrow$ **Box Refinement**$(\mathcal{L}_{S_2}, q^1, q^2, \bar{p}, \delta)$
4:    $(x_{\min}, UB) \leftarrow$ **Local Search and Adaptive Discarding**$(\mathcal{L}_{S_3}, \mathcal{A}, \delta)$
5: **end if**

---

bounds q^1 and q^2 and, thus, $\bar{p}$ changes during Algorithm 1 until line 26 and come closer to 0 in their second component, some boxes (which are already in the solution list) would be discarded if they were considered later. The filtering procedure presented in Algorithm 3 aims to find such boxes in order to delete them from the solution list $\mathcal{L}_S$.

Algorithm 3 Filtering

INPUT: (P_2), $\mathcal{L}_S$, q^1, q^2, $\bar{p}$
OUTPUT: $\mathcal{L}_{S_2}$
1: $\mathcal{L}_{S_2} \leftarrow \emptyset$
2: **while** $\mathcal{L}_S \neq \emptyset$ **do**
3:    Select a pair $(X^*, \mathcal{H})$ from $\mathcal{L}_S$
4:    **if** $\bar{p}$ is outside of outer approximation $\mathcal{H}$ of $f(X^*)$ **then**
5:       Discard X^*
6:    **else**
7:       Solve $(P^*_{X^*, \bar{p}})$ with minimal solution $(\tilde{x}, \tilde{t})$
8:       **if** $\tilde{t} > 0$ **then**
9:          Discard X^*
10:      **else** Store $(X^*, \tilde{x}, \tilde{t})$ in $\mathcal{L}_{S_2}$
11:      **end if**
12:   **end if**
13: **end while**

5.2 Box refinement

The next procedure is a refinement procedure which consists of decreasing the box width of the boxes in the solution list. By using Lipschitz properties (recall that the objective function f is continuous and bounded on a box), we can show that we obtain thereby a collection of ε-efficient solutions for (MOP_2). For every box of the solution list we find an ε-efficient solution, see the forthcoming Lemma 11. Moreover, we even get ε-efficient points which are close to a minimal solution for (P_2) in the pre-image space.

Let $\omega(X^*)$ be the box width of $X^* = [\underline{x}, \overline{x}]$, i.e., $\omega(X^*) = \|\overline{x} - \underline{x}\|$, where $\| \cdot \|$ is the Euclidean norm. During the refinement a more sophisticated rule to store boxes in the solution list is used to get an additional preciseness in the pre-image space and to ensure the ε-efficiency of the returned solutions:

2. Termination rule: *Solve* $(P^*_{X^*, \bar{p}})$ *for the box* X^* *and* $\bar{p} = (q_1^2, q_2^1)^T$, *where* q^1 *and* q^2 *are defined as in* (3) *and* (4)*. The minimal solution is denoted by* $(\tilde{x}, \tilde{t})$*. Let* $\varepsilon > 0$ *and* $\delta > 0$ *be given. Store* X^* *in* $\mathcal{L}_S$ *if*

(i) $\omega(X^*) \leq \delta$ *and*

(ii) $0 \geq \tilde{t} \geq -\frac{\varepsilon}{2}$ *and*

(iii) $(f(\tilde{x}), G(\tilde{x}))^T \lneqq \bar{p}$ *or* $\omega(X^*) < \sqrt{\frac{\varepsilon}{\max\{\alpha, \max\limits_{k=1,\dots,r} \beta_k\}}}$.

We summarize the proposed steps in Algorithm 4.

Algorithm 4 Box Refinement

INPUT: (P_2), list $\mathcal{L}_{S_2}$ of boxes a the minimal solution $(\tilde{x}, \tilde{t})$ of $(P^*_{X^*, \bar{p}})$, $q^1, q^2, \bar{p}, \delta$
OUTPUT: $\mathcal{L}_{S_3}, \mathcal{A}$
1: $\mathcal{L}_{S_3} \leftarrow \emptyset, \mathcal{A} \leftarrow \emptyset$
2: **while** $\mathcal{L}_{S_2} \neq \emptyset$ **do**
3:    Select a triple $(X^*, \tilde{x}, \tilde{t})$ from $\mathcal{L}_{S_2}$ and delete it from $\mathcal{L}_{S_2}$
4:    **if** $\tilde{t} > 0$ **then**
5:       Discard X^*
6:    **else if** $(\omega(X^*) < \delta)$ and $\left(h(\tilde{x}) \lneqq \bar{p} \text{ or } \omega(X^*) < \sqrt{\frac{\varepsilon}{\max\{\alpha, \max_{k=1,\dots,r} \beta_k\}}} \right)$ **then**
7:       Store $(X^*, \tilde{x})$ in $\mathcal{L}_{S_3}$ and $\tilde{x}$ in $\mathcal{A}$
8:    **else**
9:       Bisect X^* perpendicularly to a direction of maximum width $\rightarrow X^1, X^2$
10:       **for** $l = 1, 2$ **do**
11:          Solve $(P^*_{X^*, \bar{p}})$ with minimal solution $(\tilde{x}, \tilde{t})$
12:          Store $(X^l, \tilde{x}, \tilde{t})$ in $\mathcal{L}_{S_2}$
13:       **end for**
14:    **end if**
15: **end while**

Let L be the Lipschitz constant for f on X. A bound for L can be calculated, for example, by using interval arithmetic for the partial derivatives of f. Indeed, as boxes with minimal solutions are not discarded and for each box a representative $\tilde{x}$ is stored in $\mathcal{A}$, a bound for the minimal value f^* for (P_1) can be derived. Using Lipschitz properties for f and condition (i) from the second termination rule, we obtain $\min \{ f(x) - L\delta \mid x \in \mathcal{A} \}$ as a lower bound for f^*.

Lemma 10 *The Box Refinement, see Algorithm* 4, *needs finitely many subdivisions in line* 9.

Proof A box X^* with $\omega(X^*) < \min \left\{ \delta, \sqrt{\varepsilon / \max\{\alpha, \max_{k=1,\dots,r} \beta_k\}} \right\} := \delta'$ is stored in $\mathcal{L}_{S_3}$ automatically. Therefore, all boxes from $\mathcal{L}_{S_2}$ and their subboxes are either discarded, stored in $\mathcal{L}_{S_3}$ or bisected until their box width is smaller than δ'.   □

We omit the proof of the following lemma as it follows the structure of the proofs of Lemma 4.10 and Lemma 4.11 of [24].

Lemma 11 *Let $\mathcal{A}$ be the output of Algorithm* 4. *Then every $\tilde{x} \in \mathcal{A}$ is ε-efficient for (MOP_2).*

5.3 Local search with adaptive discarding

It is possible to further improve the outcome of Algorithm 1 with a local search algorithm which should be applied after Box Refinement, see Algorithm 4.

As we have proven in Lemmas 4 and 8, the pre-image of q^2 is ε-minimal for (P_2) if $q_2^1 > \frac{\varepsilon}{2}$ holds. In the other case we cannot guarantee any ε-minimality for the pre-image of q^2. Figure 3 shows an example, where q^2 lies far away from the minimal value f^*.

For the local search, we use each solution of $\mathcal{A}$, which came from the Box Refinement, see Algorithm 4, as a suitable starting point for a local solver to solve (P_1). Here, we solve (P_1) instead of (P_2) to avoid the handling of the max-term in (P_2). At least one of these starting points is in a δ-neighborhood of a minimal solution of (P_1). The procedure is described in Algorithm 5. Additionally, we can skip the local search procedure for some boxes in case in which the lower bound of f on a current box, which is given by $f(\tilde{x}) - L\delta$, is larger than a current upper bound for f^*. There, $\tilde{x}$ is the ε-efficient solution belonging to a box from the previous solution list and L is the Lipschitz constant of f.

Algorithm 5 Local Search for (P_1) with Adaptive Discarding

INPUT: (P_1), a list $\mathcal{L}_{S_3}$ of boxes of box width less than δ and one ε-efficient solution for each box, $\mathcal{A}$, δ
OUTPUT: $x_{\min}, UB$
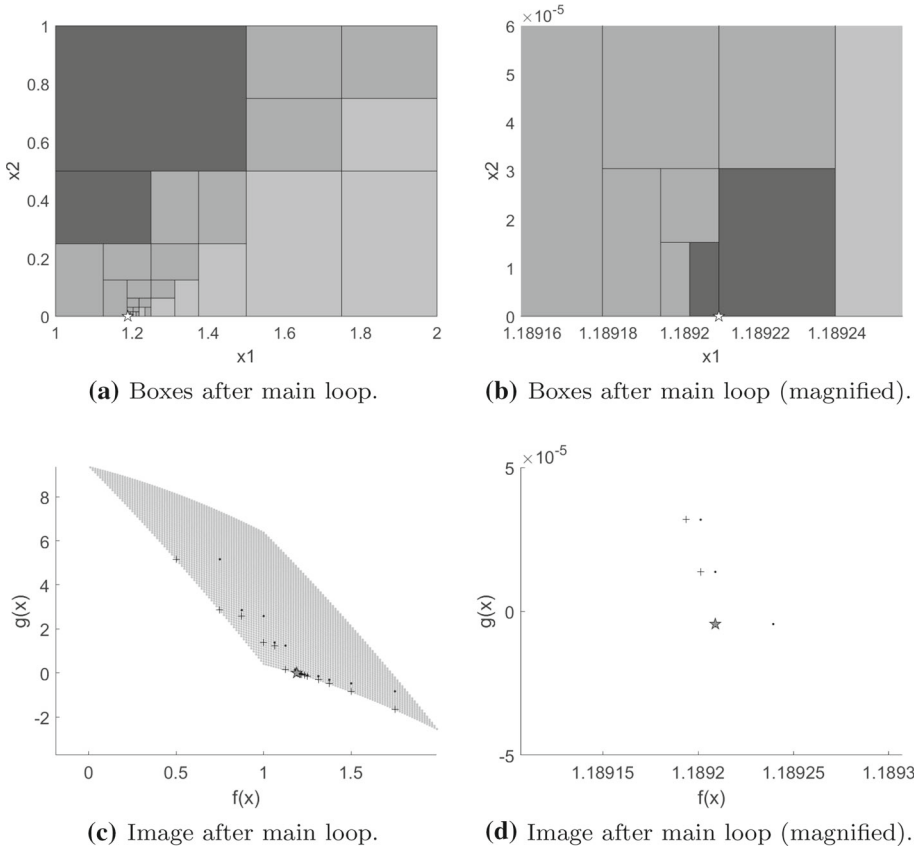1: **if** $\mathcal{A} \cap S \neq \emptyset$ **then**
2: $\quad x_{\min} \leftarrow \operatorname{argmin}_{s=1,\dots,|\mathcal{L}_{S_3}|} \left\{ f(x^s) \mid x^s \in S \right\}$
3: $\quad UB \leftarrow \min_{s=1,\dots,|\mathcal{L}_{S_3}|} \left\{ f(x^s) \mid x^s \in S \right\}$
4: **else** $x_{\min} \leftarrow \emptyset, UB \leftarrow \infty$
5: **end if**
6: **while** $\mathcal{L}_{S_3} \neq \emptyset$ **do**
7: $\quad$ Select a pair $(\tilde{X}, \tilde{x})$ from $\mathcal{L}_{S_3}$ with $\tilde{x} \in \operatorname{argmin}_{x \in \mathcal{A}}\{f(x)\}$ and delete it from $\mathcal{L}_{S_3}$
8: $\quad$ **if** $UB < f(\tilde{x}) - L\delta$ **then**
9: $\quad\quad$ Discard $\tilde{X}$
10: $\quad$ **else**
11: $\quad\quad$ Apply a local solver with starting point $\tilde{x}$ to (P_1) and obtain locally minimal solution $x^* \in S$
12: $\quad\quad x_{\min} \leftarrow \operatorname{argmin}\{f(x_{\min}), f(x^*)\}$
13: $\quad\quad UB \leftarrow \min\{UB, f(x^*)\}$
14: $\quad$ **end if**
15: **end while**

The local search strategy can also be applied if the Box Refinement is not done before. In this case the Adaptive Discarding is not possible because the boxes are not small enough. Nevertheless, a local search from an ε-efficient starting point can always be done for every box of the solution list $\mathcal{L}_{S_2}$

6 Numerical results

In this section, we examine the performance of the new algorithm MOCPαBB on Example 1 and on several additional test instances. First, all steps of the algorithm including the post processing are presented in detail for Example 1. Some of the further test instances are scalable in the number of variables and constraints. The algorithm was implemented in MATLAB R2018a. All experiments have been done on a computer with Intel(R) Core(TM) i5-7400T CPU and 16 GBytes RAM on operating system WINDOWS 10 ENTERPRISE. For the local search step, we obtain the Lipschitz constant by interval arithmetic using Intlab, [26]. For every box X^* a possible Lipschitz constant of f on X^* is $L = \sup(\|\nabla F(X^*)\|)$, where ∇F is the natural interval extension of the gradient of f. The local search was performed with `fmincon` from Matlab using the SQP-algorithm and default parameters. As it is stated in Algorithm 5, for each selected box $\tilde{X}$ the starting point for the SQP-algorithm is the ε-efficient solution $\tilde{x} \in \tilde{X}$.

(a) Boxes after main loop.

(b) Boxes after main loop (magnified).

(c) Image after main loop.

(d) Image after main loop (magnified).

Fig. 4 Results after main `while`-loop

6.1 Illustrative example

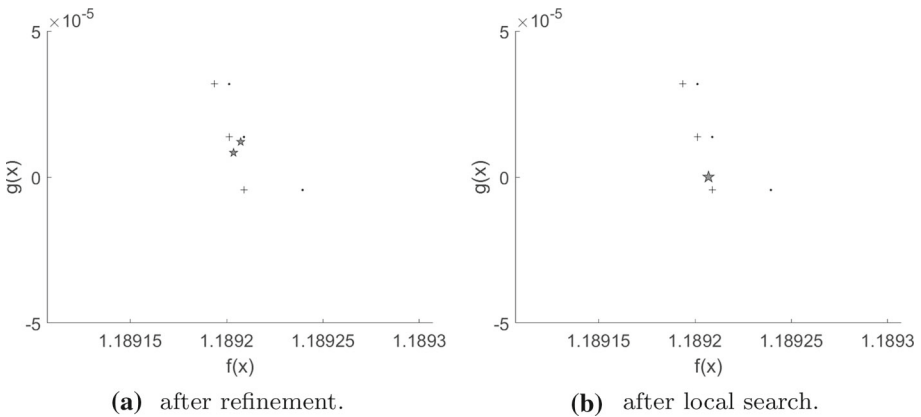First, we illustrate the new algorithm on the motivating Example 1, as given in (1), cf. [17].

We have chosen $\varepsilon = 10^{-5}$ and $\delta = 0.0001$. In the following figures, the star marks always the best found minimal solution for the single-objective problem or its image w.r.t. the multiobjective counterpart. The figures representing the pre-image space always show box partitions. The dark gray boxes are the boxes in the current solution list, i.e., those boxes, which could not yet be discarded. The light gray boxes are the discarded boxes. The different color shades represent different criteria based on which the boxes have been discarded.

In the image space, we plot some representatives of the image set by light gray points. They were obtained by discretizing the initial box and taking the function values. These points serve just for illustrative purposes on the structure of the multiobjective counterpart. The black crosses are the points of $\mathcal{L}_{PNS}$. In the magnified picture, the first cross with positive second component is q^1, and the first with negative second component is q^2. The black point above q^2 and on the right hand of q^1 is $\bar{p}$.

Figure 4 shows the box partition and the image space and their magnifications after the execution of the main `while`-loop of Algorithm 1 without the post processing step. The lightest gray boxes are the ones which got discarded by the discarding test based on Lemma 7.

(a) after filtering.   **(b)** after refinement (magnified).
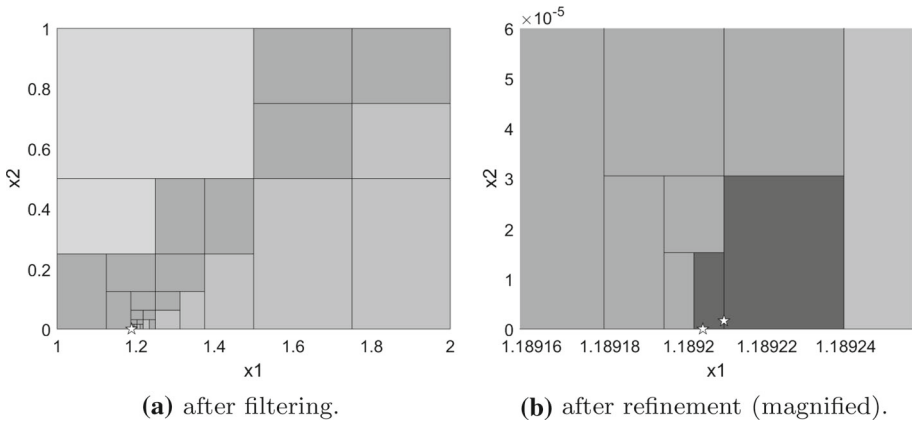
Fig. 5 Box partition after post processing—filtering and refinement

(a) after refinement.   **(b)** after local search.

Fig. 6 Image space after post processing (magnified)

The middle gray boxes are discarded because of Lemma 6. Here, it holds that $q_2^1 > \frac{\varepsilon}{2}$. Hence, with Lemma 8 the pre-image of q^2 is already $(\frac{\varepsilon}{2} + \mu)$-efficient for all $\mu > 0$. Therefore, the post processing would perform only the filtering step. For illustration, we also present here the other post processing steps.

Figure 5a illustrates the box partition of the feasible set after the first post processing step—the filtering, see Algorithm 3. The two large boxes on the upper left were discarded during this step. Concerning the minimal solution of Example 1 no improvement was obtained, and q^2 remains the best found image point so far.

In Figs. 5b and 6a the second step of the post processing is illustrated. In this step, see Algorithm 4, the boxes are usually refined until they are small enough and contain an ε-efficient solution. For this example, the two boxes are already small enough. The star shaped markers are the ε-efficient solutions and their images.

The last step of the post processing is a local search. In Fig. 6b, the star marks the image of the solution found by the local search steps. We can see that the minimal value improved slightly, because the star is a bit more on the left of q^2.

Table 1 Numbers of iterations and computational time for each step of MOPCPαBB

	# it	# sb	t [s]	$q_1^2 =$ best f^*
Main	64	5	3.0465	1.1892
Filter	5	2	0.0193	
Refine	2	2	0.0058	
Local search	2	2	0.0212	1.1892
Total	73	2	3.0928	

Table 1 states the number of iterations (# it), the number of boxes in the respective solution list (# sb) and the computational time (t) for each step. After the main loop and the local search step a feasible solution is found with a (nearly) minimal value. Those values are stated in the last column. Here, q_1^2 and the minimal value obtained by the local search step are nearly the same. Actually, the difference between both values is 1.87×10^{-6}, and the difference of the computed minimal value at the end to the actual minimal value $\sqrt[4]{2}$ is 0 (within the tolerances of MATLAB).

6.2 Test instances

In addition to Example 1, we also tested some more instances. The instance KSS2Con is as in Example 1, but with one additional constraint $x_1 + x_2 \leq 2$:

Test instance 5 *(KSS2Con) The dimension of the pre-image space is $n = 2$, and we have $r = 2$ constraints.*

$$\begin{aligned} \min \ & f(x) = x_1 - x_2 \\ \text{s.t. } & g_1(x) = -x_1^2 - (x_2 - 5)^2 + 25 + \sqrt{2} \leq 0, \\ & g_2(x) = x_1 + x_2 - 2 \leq 0, \\ & x \in [1, 2] \times [0, 1]. \end{aligned}$$

For HimmCon, which has as objective function the Himmelblau function, see [16], we added one to two constraints which exclude some of the known minima.

Test instance 6 *(HimmCon) The dimension of the preimage space is $n = 2$, and we have $r \in \{1, 2\}$ constraints.*

$$\begin{aligned} \min \ & f(x) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2 \\ \text{s.t. } & g_1(x) = -((x_1 + 3.5)^2 + (x_2 + 3.5)^2) + 4 \leq 0, \\ & g_2(x) = -(x_1 + 3)^2 - x_2 + 4 \leq 0 \ \text{only if } r = 2 \\ & x \in [-5, 5] \times [-5, 5]. \end{aligned}$$

The objective function minimized just with respect to the box constraints is known to have four globally minimal solutions with the minimal value 0:

$$x^1 = (3, 2)^T, \qquad\qquad x^2 = (-2.805118, 3.131312)^T,$$
$$x^3 = (-3.779310, -3.283186)^T, \ x^4 = (3.584428, -1.848126)^T.$$

The first added nonconvex constraint excludes the minimal solution x^3. The second nonconvex constraint excludes x^2.

The instance FF is a typical test problem for biobjective optimization, see [13]. Since the second objective has an image within the interval [0, 1] but will serve as a constraint now, we adapted this function slightly.

Test instance 7 *(FF) The dimension of the preimage space $n \in \mathbb{N}$ can be arbitrarily chosen. We have $r = 1$.*

$$\min \quad 1 - \exp\left(-\sum_{i=1}^{n}\left(x_i - \tfrac{1}{\sqrt{n}}\right)^2\right)$$
$$\text{s.t.} \quad 0.5 - \exp\left(-\sum_{i=1}^{n}\left(x_i + \tfrac{1}{\sqrt{n}}\right)^2\right) \le 0, \qquad x \in [-2, 2]^n \tag{13}$$

Since the analytical form of the efficient and nondominated set of the original biobjective optimization problem is known, we can state the minimal solution and the minimal value by

$$x^* = (1/\sqrt{n})\left(-1 + \sqrt{-\ln 0.5}\right) e \text{ and } f(x^*) \approx 0.744089815$$

Recall that e is the all-ones vector.

Moreover, we make use of the multiobjective test instance suite DTLZ, in particular of DTLZ2 and DTLZ7, [5]. These instances can be scaled regarding the number of variables and the number of objectives. Thus, the first objective is always the objective function and all other original objectives are the constraints. Again, we had to adapt the functions slightly to ensure that there are some feasible solutions.

Test instance 8 *(DTLZ2) The dimension of the preimage space $n \in \mathbb{N}$ and the number $r < n$ of constraints can be chosen arbitrarily.*

$$\min \ (1 + p(x)) \prod_{i=1}^{r} \cos(0.5\pi x_i)$$
$$\text{s.t.} \ (1 + p(x)) \sin(0.5\pi x_{r-k+1}) \prod_{i=1}^{r-k} \cos(0.5\pi x_i) \le 0.5 \text{ for all } k = 1, \ldots, r \tag{14}$$
$$x \in [0, 1]^n$$

with $p(x) = \sum_{i=r+1}^{n}(x_i - 0.5)^2$.

Test instance 9 *(DTLZ7) The dimension of the preimage space $n \in \mathbb{N}$ and the number $r < n$ of constraints can be chosen arbitrarily.*

$$\min \ x_1$$
$$\text{s.t.} \ x_{k+1} - 0.8 \le 0 \text{ for all } k = 1, \ldots, r - 1$$
$$(1 + p_1(x))p_2(x) \le r + 2, \qquad x \in [0, 1]^n \tag{15}$$

with

$$p_1(x) = 1 + \frac{9}{n-r} \sum_{i=r+1}^{n} x_i, \qquad p_2(x) = r + 1 - \sum_{i=1}^{r} \frac{x_i}{1 + p_1(x)}(1 + \sin(3\pi x_i)).$$

6.3 Numerical results

For all instances we set $\varepsilon = \delta = 0.01$ and a time limit of 6 h (21,600 s). Note that for all the instances in Table 2, the main `while`-loop took a maximum of 3969 s and most instances have been solved much faster.

Table 2 shows the overall results for all chosen test instances which were obtained within the time limit. In the first columns, we state the number n of variables and the number r of constraints. Note that for $r \ge 2$, the second objective of the multiobjective counterpart is the maximum function of all constraints. The next block shows the computational time of

Table 2 Results of MOCPαBB on different test instances

Name	n	r	t_{main}	#it$_{main}$	PP	t_{post}	#it$_{fi}$	#it$_{rf}$	#it$_{ls}$	q_1^2	best f^*	f_{min}
KSS2Con	2	2	2.38	28	1	0.06	6	2	2	1.1914	1.1892	$\approx$ 1.1892
HimmCon	2	1	4.06	51	1	129	1	37	1	2.0e−14	6.7e−13	0
	2	2	5.99	46	0	0.03	3	–	–	3.3e−14	3.3e−14	0
FF	1	1	0.66	12	1	0.20	7	5	3	0.7467	0.7441	$\approx$ 0.7441
	2	1	4.70	79	1	8.14	17	335	107	0.7496	0.7441	$\approx$ 0.7441
	3	1	33.21	527	1	295.60	117	10226	3429	0.7467	0.7441	$\approx$ 0.7441
	4	1	219.07	3010	0	3.70	326	–	–	0.7474	0.7474	$\approx$ 0.7441
DTLZ2	2	1	5.87	55	1	5.09	22	120	40	0.8701	0.8660	$\approx$ 0.8660
	3	1	24.50	239	1	131.24	69	2736	1040	0.8701	0.8660	$\approx$ 0.8660
	3	2	23.97	155	1	23.48	31	341	74	0.7086	0.7071	$\approx$ 0.7071
	4	1	113.26	999	1	11405.79	264	111424	48696	0.8701	0.8660	$\approx$ 0.8660
	4	2	116.33	665	1	722.43	125	8681	3044	0.7086	0.7071	$\approx$ 0.7071
	4	3	104.25	446	1	474.49	53	4357	1380	0.5133	0.5	0.5

Table 2 continued

Name	n	r	t_{main}	#it$_{main}$	PP	t_{post}	#it$_{fi}$	#it$_{rf}$	#it$_{ls}$	q_1^2	best f^*	f_{min}
DTLZ7	2	1	3.49	29	1	0.10	5	2	2	0.7148	0.7111	≈ 0.7111
	3	1	7.29	67	1	0.62	4	15	3	0.7148	0.7111	≈ 0.7111
	3	2	28.76	166	1	16.68	16	230	39	0.6365	0.6336	≈ 0.6336
	4	1	26.71	231	1	27.25	8	13	4	0.7148	0.7111	≈ 0.7111
	4	2	55.05	290	0	0.10	17	–	–	0.6364	0.6364	≈ 0.6336
	4	3	348.78	1316	0	0.14	22	–	–	0.0041	0.0041	
	5	1	110.89	816	1	2.72	9	61	11	0.7148	0.7111	≈ 0.7111
	5	2	157.90	755	1	8.85	15	109	11	0.6380	0.6336	≈ 0.6336
	5	3	389.30	1322	0	0.11	18	–	–	0.0047	0.0047	0
	6	1	571.28	3636	1	11.63	48	237	42	0.7148	0.7111	≈ 0.7111
	6	2	603.72	2574	1	1532	25	174	21	0.6394	0.6336	≈ 0.6336
	6	3	1017.97	3126	0	0.20	55	–	–	0.0046	0.0046	0
	7	1	3184.59	16273	1	25.62	43	470	63	0.7148	0.7111	≈ 0.7111
	7	2	2638.39	9916	1	53.77	88	549	79	0.6404	0.6336	≈ 0.6336
	7	3	3968.44	10429	0	0.31	40	–	–	0.0048	0.0048	0

(a) Box partition.

(b) Image space.
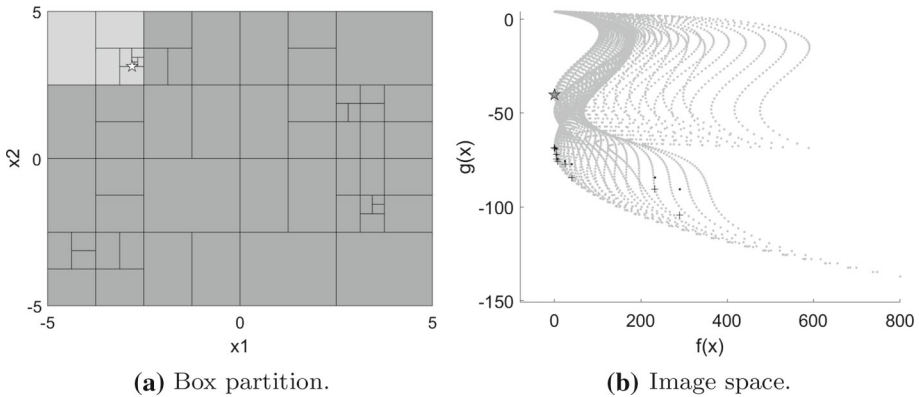
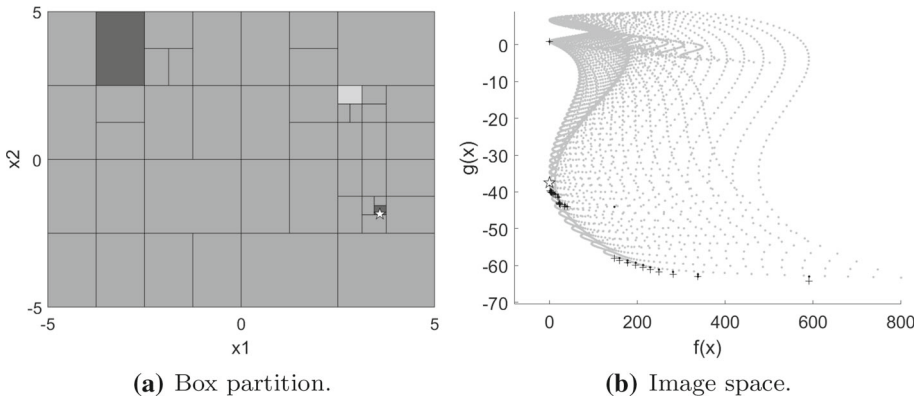**Fig. 7** Results on HimmCon with one constraint after MOCPαBB

the main `while`-loop of Algorithm 1 and the number of needed iterations. Next, the post processing is detailed. PP is the indicator whether the whole post processing is performed. If PP is 0, only the filtering was applied and because of $q_2^1 > \frac{\varepsilon}{2}$ and Lemma 8 the refinement and local search are redundant. Then the computational time and the number of needed iterations (# it$_{fi}$) are displayed. If refinement (# it$_{rf}$) and local search (# it$_{ls}$) are applied, we mention all three counts separately. The next columns state the computed nearly minimal values: First, we give q_1^2 and then the best found minimal value (best f^*). If PP$= 0$ holds, this is the value of q_1^2, since the pre-image of q^2 is always feasible. In case of PP$= 1$, the local search was done and thus, we state that minimal value which was found during the local search. To compare with the actual minimal values, we give these in the last column.

We observe that in some cases PP$= 0$ holds. The reason for that is $q_2^1 > \frac{\varepsilon}{2} = 0.005$, and by Lemma 8 the pre-image of q^2 is already ε-minimal for the single-objective optimization problem. Thus, the additional post processing, i. e., refinement and local search do not have to be executed. Even if those steps have been performed for most of the cases, we also observed that the value q_1^2 is already close to the real minimal value and therefore its pre-image is ε-efficient for (P_1). The only instance for which the pre-image of q^2 is not ε-efficient is DTLZ2 with $n = 4, r = 3$, see the italic entry of q_1^2.

We give in the following some more details on one of the instances, on HimmCon. Figures 7 and 8 illustrate the results of MOCPαBB for the cases $r \in \{1, 2\}$, i.e., with one or two constraints. The meaning of the different shades of the boxes and points are the same as in Sect. 6.1. As the run for the optimization problem with one constraint did the refinement and local search step, the minimal solution and minimal value, visualized by the stars in Fig. 7, are found during the local search. In fact, the minimal solution there is $x^* = (-2.8051, 3.1313)^T$ with $f(x^*) \approx 0$. For two constraints, $q_2^1 \approx 0.83$ was large enough to skip the refinement and local search steps. Thus, the star in Fig. 8a is the pre-image of q^2, i.e, $x^{**} = (3.5844, -1.8481)^T$, and the one in Fig. 8b is $q^2 = (0.0000, -37.5066)^T$ itself. Both found solution correspond to one of the minimal solutions of the Himmelblau function without additional constraints, i.e., $x^* \approx x^2$ and $x^{**} \approx x^4$.

6.4 Discussion

The above analysis shows that taking a multiobjective perspective on constrained optimization in general, and on αBB-methods in particular, leads to a new and promising class of

(a) Box partition.

(b) Image space.

Fig. 8 Results on HimmCon with two constraints after MOCPαBB

global optimization algorithms. An additional benefit can be seen in the fact that the solution alternatives that are generated for the multiobjective counterpart problems provide trade-off information on objective value minimization versus constraint satisfaction. This can be nicely seen at the example of the HimmCon instances in Figs. 7b and 8b.

We emphasize that our implementation is prototypical and that no state-of-the-art pre-processing was used. We can thus not expect computational times that are competitive with other (commercial) state-of-the-art solvers. Indeed, the above test instances were solved within (milli-)seconds using GAMS models [14] with BARON [33] (instances KSS2Con, HimmCon, FF) and MINOS [23] (instances DTLZ2 and DTLZ7), and using the built-in advanced pre-processing routines that in most cases already returned near-optimal solutions.

Moreover, our numerical results clearly show one of the main drawbacks of branch and bound type methods in this context: Scaling the number of variables increases the computational time. This is clear, because the branching happens in the pre-image space and for more variables a box has to be bisected more often to become small. Despite this shortcoming, bounding and discarding have proven to be highly effective so that the number of active boxes remains comparably small and storage requirements were never a problem.

7 Conclusions

Within this paper, we proposed to use a multiobjective counterpart to find feasible solutions for nonconvex single-objective constrained optimization problems. It is well-known that finding feasible solutions can be a hard task and we proposed a new approach to find such points. For that approach, we adapted a global solution method for multiobjective optimization problems such that it concentrates on the region of interest. By using such multiobjective counterparts one also gets additional information on the trade-off which one would obtain by relaxing the constraints slightly in favour of an improved objective function value.

References

1. Adjiman, C.S., Dallwig, S., Floudas, C.A., Neumaier, A.: A global optimization method, αBB, for general twice-differentiable constrained NLPs: I—theoretical advances. Comput. Chem. Eng. **22**(9), 1137–1158 (1998)
2. Campana, E.F., Diez, M., Liuzzi, G., Lucidi, S., Pellegrini, R., Piccialli, V., Rinaldi, F., Serani, A.: A multi-objective DIRECT algorithm for ship hull optimization. Comput. Optim. Appl. **71**(1), 53–72 (2018)
3. Deb, K.: Multi-objective genetic algorithms: problem difficulties and construction of test problems. Evol. Comput. **7**(3), 205–230 (1999)
4. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 181–197 (2002)
5. Deb, K., Thiele, L., Laumanns, M., Zitzler, E.: Scalable multi-objective optimization test problems. In: Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No. 02TH8600), vol. 1, pp. 825–830 (2002)
6. Ehrgott, M.: Multicriteria Optimisation, 2nd edn. Springer-Verlag, Berlin (2005)
7. Ehrgott, M., Shao, L., Schöbel, A.: An approximation algorithm for convex multi-objective programming problems. J. Global Optim. **50**(3), 397–416 (2011)
8. Eichfelder, G., Klamroth, K., Niebling, J.: Using a B&B algorithm from multiobjective optimization to solve constrained optimization problems. In: AIP Conference Proceedings, vol. 2070, p. 020028 (2019)
9. Evtushenko, Y.G., Posypkin, M.A.: A deterministic algorithm for global multi-objective optimization. Optim. Methods Softw. **29**(5), 1005–1019 (2014)
10. Fernández, J., Tóth, B.: Obtaining the efficient set of nonlinear biobjective optimization problems via interval branch-and-bound methods. Comput. Optim. Appl. **42**(3), 393–419 (2009)
11. Fletcher, R., Leyffer, S.: Nonlinear programming without a penalty function. Math. Program. **91**, 239–269 (2002)
12. Fletcher, R., Leyffer, S., Toint, P.L.: A brief history of filter methods. Technical Report ANL/MCS-P1372-0906 (2006)
13. Fonseca, C., Fleming, P.: Multiobjective genetic algorithms made easy: selection sharing and mating restriction. In: Proceedings of the 1st International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications, pp. 45–52. IEEE Press (1995)
14. GAMS Development Corporation. General Algebraic Modeling System (GAMS) Release 24.1.3. Washington, DC, USA (2013)
15. Günther, C., Tammer, C.: Relationships between constrained and unconstrained multi-objective optimization and application in location theory. Math. Methods Oper. Res. **84**(2), 359–387 (2016)
16. Himmelblau, D.M.: Applied Nonlinear Programming. McGraw-Hill, New York (1972)
17. Kirst, P., Stein, O., Steuermann, P.: Deterministic upper bounds for spatial branch-and-bound methods in global minimization with nonconvex constraints. TOP **23**(2), 591–616 (2015)
18. Klamroth, K., Tind, J.: Constrained optimization using multiple objective programming. J. Global Optim. **37**(3), 325–355 (2007)
19. Löhne, A., Rudloff, B., Ulus, F.: Primal and dual approximation algorithms for convex vector optimization problems. J. Global Optim. **60**(4), 713–736 (2014)
20. Maranas, C.D., Floudas, C.A.: Global minimum potential energy conformations of small molecules. J. Global Optim. **4**(2), 135–170 (1994)
21. Martin, B., Goldsztejn, A., Granvilliers, L., Jermann, C.: Constraint propagation using dominance in interval branch & bound for nonlinear biobjective optimization. Eur. J. Oper. Res. **260**(3), 934–948 (2016)
22. Miettinen, K.: Nonlinear Multiobjective Optimization. Springer-Verlag, Berlin (1998)
23. Murtagh, B.A., Gill, P.E., Murray, W., Saunders, M.A., Wright, M.H.: MINOS 5.51, Large Scale Nonlinear Solver (2004)
24. Niebling, J., Eichfelder, G.: A branch-and-bound-based algorithm for nonconvex multiobjective optimization. SIAM J. Optim. **29**(1), 794–821 (2019)

25. Pardalos, P., Žilinskas, A., Žilinskas, J.: Non-convex Multi-objective Optimizationn. Springer-Verlag, Berlin (2017)
26. Rump, S.M.: INTLAB—INTerval LABoratory. In: Csendes, T. (ed.) Developments in Reliable Computing, pp. 77–104. Kluwer Academic Publishers, Dordrecht (1999)
27. Sawaragi, Y., Nakayama, H., Tanino, T.: Theory of Multiobjective Optimization. Academic Press, Cambridge (1985)
28. Scholz, D.: The multicriteria big cube small cube method. TOP **18**, 286–302 (2010)
29. Scholz, D.: Deterministic Global Optimization: Geometric Branch-and-Bound Methods and Their Applications. Springer, Berlin (2012)
30. Schulze, B., Paquete, L., Klamroth, K., Figueira, J.R.: Bi-dimensional knapsack problems with one soft constraint. Comput. Oper. Res. **78**, 15–26 (2017)
31. Darup, M.S., Mönnigmann, M.: Improved automatic computation of Hessian matrix spectral bounds. SIAM J. Sci. Comput. **38**(4), A2068–A2090 (2016)
32. Segura, C., Coello, C.A.C., Miranda, G., León, C.: Using multi-objective evolutionary algorithms for single-objective constrained and unconstrained optimization. Ann. Oper. Res. **240**(1), 217–250 (2016)
33. Tawarmalani, M., Sahinidis, N.V.: A polyhedral branch-and-cut approach to global optimization. Math. Program. **103**, 225–249 (2005)
34. Žilinskas, A., Calvin, J.: Bi-objective decision making in global optimization based on statistical models. J. Global Optim. **74**, 599–609 (2019)
35. Žilinskas, A., Žilinskas, J.: Adaptation of a one-step worst-case optimal univariate algorithm of bi-objective Lipschitz optimization to multidimensional problems. Commun. Nonlinear Sci. Numer. Simul. **21**(1–3), 89–98 (2016)