Computational Optimization and Applications (2022) 82:293–327
https://doi.org/10.1007/s10589-022-00363-1

Derivative-free methods for mixed-integer nonsmooth constrained optimization

Tommaso Giovannelli[1] · Giampaolo Liuzzi[2] · Stefano Lucidi[3] · Francesco Rinaldi[4]

Received: 6 July 2021 / Accepted: 8 March 2022 / Published online: 1 April 2022

Abstract

In this paper, mixed-integer nonsmooth constrained optimization problems are considered, where objective/constraint functions are available only as the output of a black-box zeroth-order oracle that does not provide derivative information. A new derivative-free linesearch-based algorithmic framework is proposed to suitably handle those problems. First, a scheme for bound constrained problems that combines a dense sequence of directions to handle the nonsmoothness of the objective function with primitive directions to handle discrete variables is described. Then, an exact penalty approach is embedded in the scheme to suitably manage nonlinear (possibly nonsmooth) constraints. Global convergence properties of the proposed algorithms toward stationary points are analyzed and results of an extensive numerical experience on a set of mixed-integer test problems are reported.

Keywords Derivative-free optimization · Nonsmooth optimization · Mixed-integer nonlinear programming

✉ Giampaolo Liuzzi
liuzzi@diag.uniroma1.it

Tommaso Giovannelli
tog220@lehigh.edu

Stefano Lucidi
lucidi@diag.uniroma1.it

Francesco Rinaldi
rinaldi@math.unipd.it

1 Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, USA

2 Dipartimento di Ingegneria Informatica, Automatica e Gestionale, Sapienza Università di Roma and Istituto di Analisi dei Sistemi e Informatica (IASI), CNR, Roma, Italy

3 Dipartimento di Ingegneria Informatica, Automatica e Gestionale, Sapienza Università di Roma, Roma, Italy

4 Dipartimento di Matematica “Tullio Levi-Civita”, Università di Padova, Padova, Italy

Mathematics Subject Classification 90C11 · 90C56 · 65K05

1 Introduction

The following mixed-integer nonlinearly constrained problem is considered

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & f(x) \\ \text{s.t.} \quad & g(x) \le 0, \\ & l \le x \le u, \\ & x_i \in \mathbb{R} \ \text{ for all } \ i \in I^c, \\ & x_i \in \mathbb{Z} \ \text{ for all } \ i \in I^z, \end{aligned} \tag{1.1}$$

where $x \in \mathbb{R}^n$, $l, u \in \mathbb{R}^n$, and $I^c \cup I^z = \{1, 2, \dots, n\}$, with $I^c \cap I^z = \emptyset$ and $I^c, I^z \neq \emptyset$.[1] We assume $l_i < u_i$ for all $i \in I^c \cup I^z$, and $l_i, u_i \in \mathbb{Z}$ for all $i \in I^z$. Moreover, the functions $f : \mathbb{R}^n \to \mathbb{R}$ and $g : \mathbb{R}^n \to \mathbb{R}^m$, which may be nondifferentiable, are supposed to be Lipschitz continuous with respect to x_i for all $i \in I^c$, i.e., for all $x, y \in \mathbb{R}^n$ a constant $L > 0$ exists such that

$$|f(x) - f(y)| \le L\|x - y\|, \quad \text{with } \ x_i = y_i, \ \text{for all } \ i \in I^z. \tag{1.2}$$

The following sets are defined

$$X := \{x \in \mathbb{R}^n : l \le x \le u\}, \quad \mathcal{F} := \{x \in \mathbb{R}^n : g(x) \le 0\},$$
$$\mathcal{Z} := \{x \in \mathbb{R}^n : x_i \in \mathbb{Z} \text{ with } i \in I^z\}.$$

Throughout the paper X is assumed to be a compact set. Therefore, l_i and u_i cannot be infinite.

Remark 1 Note that $X \cap \mathcal{Z}$ is compact too. In fact, let us consider any sequence $\{x_k\} \subseteq X \cap \mathcal{Z}$ such that $x_k \to \bar{x}$. Since X is compact, $\bar{x} \in X$. Furthermore, for k sufficiently large, the integer components of x_k are fixed, then $\bar{x} \in \mathcal{Z}$. Hence, $\bar{x} \in X \cap \mathcal{Z}$, meaning that $X \cap \mathcal{Z}$ is compact too.

Problem (1.1) can hence be reformulated as follows

$$\begin{aligned} & \min_{x\in\mathbb{R}^n} f(x) \\ & \text{s.t.} \quad x \in \mathcal{F} \cap \mathcal{Z} \cap X. \end{aligned} \tag{1.3}$$

The objective and constraint functions in (1.3) are assumed to be of black-box zeroth-order type, which is to say that the analytical expression is unknown, and the function value corresponding to a given point is the only available information. Therefore, black-box Mixed-Integer Nonlinear Programs (MINLPs) are considered,

[1] Note that the assumption $I^c, I^z \neq \emptyset$ is not a restrictive one. Indeed, when $I^z = \emptyset$, i.e., there are no discrete variables (respectively $I^c = \emptyset$, i.e., there are no continuous variables), the theory exactly follows from [21] (respectively [35]).

a class of challenging problems frequently arising in real-world applications. Those problems are usually solved through tailored derivative-free optimization algorithms (see, e.g., [8, 12, 15, 30] and references therein for further details) able to properly manage the presence of both continuous and discrete variables.

The optimization methods for black-box MINLPs that we consider in here can be divided into two main classes: direct-search and model-based methods. The direct-search methods for MINLPs usually share two main features: they perform an alternate minimization between continuous and discrete variables, and use a fixed neighborhood to explore the integer lattice. In particular, [4] adapts the Generalized Pattern Search (GPS), proposed in [50], to solve problems with categorical variables (those variables include integer variables as a special case), so-called mixed variable problems. The approach in [4] has been then extended to address problems with general constraints [1] and stochastic objective function [49]. In [1], constraints are tackled by using a filter approach similar to the one described in [5]. Derivative-free methods for categorical variables and general constraints have also been studied in [37] and [2]. In particular, [37] proposes a general algorithmic framework whose global convergence holds for any continuous local search (e.g., a pattern search) satisfying suitable properties. The Mesh Adaptive Direct Search (MADS), originally introduced in [6] for nonsmooth problems under general constraints, is extended in [2] to solve mixed variable problems. Constraints are tackled through an extreme barrier approach in this case. The original MADS algorithm has been recently extended in [11] to solve problems with "*granular variables, i.e., variables with fixed number of decimals*", and nonsmooth objective function over the continuous variables. In addition to the previous references [1, 2, 4, 5, 11, 37, 50], another work that is worth mentioning is [45], where a mesh-based direct-search algorithm is proposed for bound constrained mixed-integer problems involving nonsmooth and noncontinuous objectives.

In [33], three algorithms are proposed for bound constrained MINLP problems. Unlike the aforementioned works, the discrete neighborhood does not have a fixed structure, but depends on a linesearch-type procedure. The first algorithm in [33] performs a distributed minimization over all the variables by updating the current iterate as soon as a point ensuring a sufficient decrease of the objective function is found. It was extended in [34], which deals with the constrained case by adopting a sequential penalty approach, and [52], where the maximal positive basis is replaced with a minimal positive basis based on a direction-rotation technique. Bound constrained MINLP problems are also considered in [23], which extends the algorithm for continuous smooth and nonsmooth objective functions introduced in [22].

Some other direct-search methods not directly connected with MINLP problems are reported for their influence on algorithm development. In [21], the authors propose a new linesearch-based method for nonsmooth nonlinearly constrained optimization problems, ensuring convergence towards Clarke-Jahn stationary points. The constraints are tackled through an exact penalty approach. In [17] and [18], the authors analyze the benefit in terms of efficiency deriving from different ways of incorporating the simplex gradient into direct-search algorithms (e.g., GPS and MADS) for minimizing objective functions which not necessarily require continuous differentiability. In [51], the authors analyze the convergence properties of direct-search methods applied to the minimization of discontinuous functions.

Model-based methods are also widely used in derivative-free optimization to solve MINLPs. In [16], the authors describe an open-source library, called RBFOpt, that uses surrogate models based on radial basis functions for handling bound constrained MINLPs. The same class of problems is also tackled in [44] through quadratic models. This paper extends to the mixed-integer case the trust-region derivative-free algorithm BOBYQA introduced in [46] for continuous problems. Surrogate models employing radial basis functions are used in [43] to describe an algorithm, called SO-MI, able to converge to global optimizers of the problem almost surely. A similar algorithm, called SO-I, is proposed by the same authors in [42] to address integer global optimization problems. In [41], the authors propose an algorithm for MINLP problems that modifies the sampling strategy used in SO-MI and uses also an additional local search. Finally, Kriging models were effectively used in [27] and [25] to develop new sequential algorithms. Models can also be used to boost direct-search methods. For example, in NOMAD, i.e., the software package that implements the MADS algorithm (see [9, 32]), a surrogate-based model is used to generate promising points.

In [31] and [35] methods for black-box problems with only unrelaxable integer variables are devised. In particular, the authors in [31] propose a method for minimizing convex black-box integer problems that uses secant functions interpolating previous evaluated points. In [35], a new method based on a nonmonotone linesearch and primitive directions is proposed to solve a more general problem where the objective function is allowed to be nonconvex. The primitive directions allow the algorithm to escape bad local minima, thus providing the potential to find a global optimum, even if this typically requires the exploration of large neighborhoods.

In this paper, new derivative-free linesearch-type algorithms for mixed-integer nonlinearly constrained problems with possibly nonsmooth functions are proposed. The strategies successfully tested in [21] and [35] for continuous and integer problems, respectively, are combined to devise a globally convergent algorithmic framework that allows tackling the mixed-integer case. Continuous and integer variables are suitably handled by means of specific local searches in this case. On the one hand, a dense sequence of search directions is used to explore the subspace related to the continuous variables and detect descent directions, whose cone can be arbitrarily narrow due to nonsmoothness. On the other hand, a set of primitive discrete directions is adopted to guarantee a thorough exploration of the integer lattice in order to escape bad local minima. A first algorithm for bound constrained problems is developed, then it is adapted to handle the presence of general nonlinear constraints by using an exact penalty approach. Since only the violation of such constraints is included in the penalty function, the algorithm developed for bound constrained problems can be easily adapted to minimize the penalized problem.

With regard to the convergence results, it can be proved that particular sequences of iterates yielded by the two algorithms converge to suitably defined stationary points of the problem considered. In the generally constrained case, this result is based on the equivalence between the original problem and the penalized problem.

The paper is organized as follows. In Sect. 2, we report some definitions and preliminary results. In Sect. 3, we describe the algorithm proposed for mixed-integer problems with bound constraints and we analyze its convergence properties. The same type of analysis is reported in Sect. 4 for the algorithm addressing mixed-integer problems with general nonlinear constraints. Section 5 describes the results of extensive numerical experiments performed for both algorithms. Finally, in Sect. 6 we include some concluding remarks and we discuss future work.

2 Notation and preliminary results

Given a vector $v \in \mathbb{R}^n$, we introduce the subvectors $v_c \in \mathbb{R}^{|I^c|}$ and $v_z \in \mathbb{R}^{|I^z|}$, given by

$$v_c = \left[v_i\right]_{i \in I^c} \quad \text{and} \quad v_z = \left[v_i\right]_{i \in I^z},$$

where v_i denotes the i-th component of v. When a vector is an element of an infinite sequence of vectors $\{v_k\}$, the i-th component will be denoted as $(v_k)_i$, in order to avoid possible ambiguities. Moreover, throughout the paper we denote by $\|\cdot\|$ the Euclidean norm.

The search directions considered in the algorithms proposed in the next sections have either a null continuous subvector or a null discrete subvector, meaning that we do not consider directions that update both continuous and discrete variables simultaneously. We first report the definition of primitive vector, used to characterize the subvectors of the search directions related to the integer variables. Then we move on to the properties of the subvectors related to the continuous variables.

From [35] we report the following definition of primitive vector.

Definition 1 (*Primitive vector*) A vector $v \in \mathbb{Z}^n$ is called primitive if the greatest common divisor of its components $\{v_1, v_2, \dots, v_n\}$ is equal to 1.

Since the objective and constraint functions of the problem considered are assumed to be nonsmooth when fixing the discrete variables, proving convergence to a stationary point requires particular subsequences of the continuous subvectors of the search directions to be provided with the density property. Since the feasible descent directions can form an arbitrarily narrow cone (see, e.g., [3] and [6]), a finite number of search directions is indeed not sufficient. The unit sphere with respect to the continuous variables with center in the origin is denoted as

$$S(0,1) \triangleq \{s \in \mathbb{R}^n \ : \ \|s_c\| = 1 \text{ and} \|s_z\| = 0\}.$$

Then, the definition of a dense subsequence of directions given in [21] is extended to the mixed-integer case.

Definition 2 (*Dense subsequence*) Let K be an infinite subset of indices (possibly $K = \{0, 1, \dots\}$) and $\{s_k\} \subset S(0,1)$ a given sequence of directions. The subsequence

$\{s_k\}_K$ is said to be dense in $S(0, 1)$ if, for any $\bar{s} \in S(0,1)$ and for any $\epsilon > 0$, there exists an index $k \in K$ such that $\|s_k - \bar{s}\| \leq \epsilon$.

Similarly to what is done in [2], the definition of generalized directional derivative, which is also called Clarke directional derivative, given in [14] is extended to the mixed-integer case. This allows providing necessary optimality conditions for Problem (1.3). We also recall the definition of generalized gradient.

Definition 3 (*Generalized directional derivative and generalized gradient*) Let $h : \mathbb{R}^n \to \mathbb{R}$ be a Lipschitz continuous function near $x \in \mathbb{R}^n$ with respect to its continuous variables x_c [see, e.g., (1.2)]. The generalized directional derivative of h at x in the direction $s \in \mathbb{R}^n$, with $s_i = 0$ for $i \in I^z$, is

$$h_c^{Cl}(x;s) = \limsup_{y_c \to x_c, y_z = x_z, t \downarrow 0} \frac{h(y + ts) - h(y)}{t}. \tag{2.1}$$

To simplify the notation, the generalized gradient of h at x w.r.t the continuous variables can be redefined as

$$\begin{aligned}\partial_c h(x) =\{& v \in \mathbb{R}^n \ : \ v_i = 0, \ i \in I^z, \ \text{ and } h_c^{Cl}(x;s) \geq s^T v \ \text{ for all } s \in \mathbb{R}^n, \\ & \text{with } s_i = 0 \text{ for } i \in I^z\}.\end{aligned}$$

Moreover, let us denote the orthogonal projection over the set X as $[x]_{[l,u]} = \max\{l, \min\{u, x\}\}$ and the interior of a set $\mathcal{C}$ as $\mathring{\mathcal{C}}$. These concepts will be used throughout the paper.

2.1 The bound constrained case

First, a simplified version of Problem (1.1) is considered, where only bound constraints are present in the definition of the feasible set. The resulting problem will also allow tackling the nonlinearly constrained case. An exact penalty approach is indeed adopted to deal with the general nonlinear constraints, thus giving rise to a bound constrained problem in the end. In particular, the following formulation is considered in this section:

$$\begin{aligned}\min_{x\in\mathbb{R}^n} \ & f(x) \\ \text{s.t.} \ & l \leq x \leq u, \\ & x_i \in \mathbb{R} \ \text{ for all } \ i \in I^c, \\ & x_i \in \mathbb{Z} \ \text{ for all } \ i \in I^z.\end{aligned}$$

Such a problem can be reformulated as follows

$$\begin{aligned}&\min_{x\in\mathbb{R}^n} f(x) \\ &\text{s.t.} \quad x \in X \cap \mathcal{Z}.\end{aligned} \tag{2.2}$$

When considering Problem (2.2), the basic concept of *feasible direction* must be specialized to take into proper account the presence of discrete variables along with continuous ones. In particular, for discrete variables, primitive directions can be used to define the set of feasible directions. We hence suitably adapt the definition of feasible primitive direction set given in [35] to the mixed-integer case.

Definition 4 (*Set of feasible primitive directions*) Given a point $x \in X \cap \mathcal{Z}$,

$$D^z(x) = \{d \in \mathbb{Z}^n \,:\, d_z \text{ is a primitive vector, } d_i = 0 \text{ for all } i \in I^c, \textit{ and } \\ x + d \in X \cap \mathcal{Z}\}$$

is the set of feasible primitive directions at x with respect to $X \cap \mathcal{Z}$.

We introduce two definitions of neighborhood related to the discrete variables and we recall the definition of neighborhood related to the continuous variables. They are used to formally define local minimum points.

Definition 5 (*Discrete neighborhood*) Given a point $\bar{x} \in X \cap \mathcal{Z}$, the discrete neighborhood of $\bar{x}$ is

$$\mathcal{B}^z(\bar{x}) = \{x \in X \cap \mathcal{Z} : \quad x = \bar{x} + d, \quad \text{with} \quad d \in D^z(\bar{x})\}.$$

Definition 6 (*Continuous neighborhood*) Given a point $\bar{x} \in X \cap \mathcal{Z}$ and a scalar $\rho > 0$, the continuous neighborhood of $\bar{x}$ is

$$\mathcal{B}^c(\bar{x};\rho) = \left\{x \in X \,:\, x_z = \bar{x}_z \text{ and } \|x_c - \bar{x}_c\| \le \rho\right\}.$$

Then, the definition of local minimum points for the bound constrained Problem (2.2) is reported. Basically, a point is referred to as a local minimum when:

(i) it is a local minimum w.r.t. the continuous variables;
(ii) it is a local minimum w.r.t. its discrete neighborhood.

Definition 7 (*Local minimum point*) A point $x^* \in X \cap \mathcal{Z}$ is a local minimum point of Problem (2.2) if, for some $\epsilon > 0$,

$$f(x^*) \le f(x) \quad \text{for all } x \in \mathcal{B}^c(x^*;\epsilon), \tag{2.3}$$

$$f(x^*) \le f(x) \quad \text{for all } x \in \mathcal{B}^z(x^*). \tag{2.4}$$

Now, taking into account the presence of the bound constraints in Problem (2.2), the cone of feasible continuous directions is introduced. This set is used to define stationary points for Problem (2.2).

Definition 8 (*Cone of feasible continuous directions*) Given a point $x \in X \cap \mathcal{Z}$, the set

$$\begin{aligned} D^c(x) = \{s \in \mathbb{R}^n : s_i = 0 \quad &\text{for all} \quad i \in I^z, \\ s_i \geq 0 \quad \text{for all} \quad i \in I^c \quad &\text{and} \quad x_i = l_i, \\ s_i \leq 0 \quad \text{for all} \quad i \in I^c \quad &\text{and} \quad x_i = u_i, \\ s_i \in \mathbb{R} \quad \text{for all} \quad i \in I^c \quad &\text{and} \quad l_i < x_i < u_i, \} \end{aligned}$$

is the cone of feasible continuous directions at x with respect to $X \cap \mathcal{Z}$.

Now, the definition of Clarke–Jahn generalized directional derivative given in [28, Section 3.5] is extended to the mixed-integer case. As opposed to the Clarke directional derivative, in this definition the limit superior is considered only for points y and $y + ts$ in $X \cap \mathcal{Z}$, thus requiring stronger assumptions.

Definition 9 (*Clarke–Jahn generalized directional derivative*) Given a point $x \in X \cap \mathcal{Z}$ with continuous subvector x_c, the Clarke–Jahn generalized directional derivative of function f along direction $s \in D^c(x)$ is given by:

$$f_c^\circ(x;s) = \limsup_{\substack{y_c \to x_c, y_z = x_z, y \in X \cap \mathcal{Z} \\ t \downarrow 0, y + ts \in X \cap \mathcal{Z}}} \frac{f(y+ts) - f(y)}{t} \tag{2.5}$$

Finally, a few basic stationarity definitions are reported which will be used in the convergence analysis. More specifically, it will be proved that limit points of the sequence generated by the proposed algorithmic framework exist which are stationary for Problem (2.2).

Definition 10 (*Stationary point*) A point $x^* \in X \cap \mathcal{Z}$ is a stationary point of Problem (2.2) when

$$f_c^\circ(x^*;s) \geq 0, \qquad \text{for all} \quad s \in D^c(x^*), \tag{2.6}$$

$$f(x^*) \leq f(x), \qquad \text{for all} \quad x \in \mathcal{B}^z(x^*). \tag{2.7}$$

Definition 11 (*Clarke stationary point*) A point $x^* \in X \cap \mathcal{Z}$ is a Clarke stationary point of Problem (2.2) when it satisfies

$$f_c^{Cl}(x^*;s) \geq 0 \qquad \text{for all} \quad s \in D^c(x^*), \tag{2.8}$$

$$f(x^*) \leq f(x) \qquad \text{for all} \quad x \in \mathcal{B}^z(x^*). \tag{2.9}$$

2.2 The nonsmooth nonlinearly constrained case

In this subsection, Problem (1.3) is considered. A local minimum point for this problem is defined as follows.

Definition 12 (*Local minimum point*) A point $x^* \in \mathcal{F} \cap \mathcal{Z} \cap X$ is a local minimum point of Problem (1.3) if, for some $\epsilon > 0$,

$$\begin{aligned} f(x^*) &\le f(x) \text{ for all } x \in \mathcal{B}^c(x^*;\epsilon) \cap \mathcal{F}, \\ f(x^*) &\le f(x) \text{ for all } x \in \mathcal{B}^z(x^*) \cap \mathcal{F}. \end{aligned} \tag{2.10}$$

Exploiting the necessary optimality conditions introduced in [21], the following KKT stationarity definition can be stated. This definition will also be used in the convergence analysis. More specifically, it will be proved that, using a simple penalty approach, limit points of the sequence generated by the proposed algorithmic framework exist which are KKT stationary for Problem (1.3).

Definition 13 (*KKT stationary point*) A point $x^* \in \mathcal{F} \cap \mathcal{Z} \cap X$ is a KKT stationary point of Problem (1.3) if there exists a vector $\lambda^* \in \mathbb{R}^m$ such that, for every $s \in D^c(x^*)$,

$$\max \left\{ \xi^\top s \ : \ \xi \in \partial_c f(x^*) + \sum_{i=1}^m \lambda_i^* \partial_c g_i(x^*) \right\} \ge 0, \tag{2.11}$$

$$(\lambda^*)^T g(x^*) = 0 \ \text{ and } \ \lambda^* \ge 0, \tag{2.12}$$

and

$$f(x^*) \le f(x) \ \text{ for all } \ x \in \mathcal{B}^z(x^*) \cap \mathcal{F}. \tag{2.13}$$

3 An algorithm for bound constrained problems

In this section, an algorithm for solving the mixed-integer bound constrained problem defined in Problem (2.2) is proposed, and its convergence properties are analyzed. The optimization over the continuous and discrete variables is performed by means of two local searches based on linesearch algorithms that explore the feasible search directions similarly to the procedures proposed in [21, 33, 34, 36]. In particular, the *Projected Continuous Search* described in Algorithm 1 and the *Discrete Search* described in Algorithm 2 are the methods adopted to investigate the directions associated with the continuous and discrete variables, respectively. The idea behind the line searches is to return a positive stepsize α, namely to update the current iterate, whenever a point providing a sufficient reduction of the objective function is found. In Algorithm 1 the sufficient decrease is controlled by the parameter α, while in Algorithm 2 the same role is played by ξ. Once such a point is determined, an expansion step is performed in order to explore if the sufficient reduction may be achieved through a larger stepsize.

Algorithm 1 Projected Continuous Search $(\tilde{\alpha}, w, p; \alpha, \tilde{p})$

```
Data. $\gamma > 0$, $\delta \in (0,1)$.

Step 0. Set $\alpha = \tilde{\alpha}$.
Step 1. If $f([w + \alpha p]_{[l,u]}) \le f(w) - \gamma\alpha^2$ then set $\tilde{p} = p$ go to Step 4.
Step 2. If $f([w - \alpha p]_{[l,u]}) \le f(w) - \gamma\alpha^2$ then set $\tilde{p} = -p$ and go to Step 4.
Step 3. Set $\alpha = 0$, return $\alpha$ and $\tilde{p} = p$.
Step 4. Let $\beta = \alpha/\delta$.
Step 5. If $f([w + \beta\tilde{p}]_{[l,u]}) > f(w) - \gamma\beta^2$ return $\alpha$, $\tilde{p}$.
Step 6. Set $\alpha = \beta$ and go to Step 4.
```

Algorithm 2 Discrete Search $(\tilde{\alpha}, w, p, \xi; \alpha)$

```
Data. $\gamma > 0$.

Step 0. Compute the largest $\bar{\alpha}$ such that $w + \bar{\alpha}p \in X \cap \mathcal{Z}$.
        Set $\alpha = \min\{\bar{\alpha}, \tilde{\alpha}\}$.
Step 1. If $\alpha > 0$ and $f(w + \alpha p) \le f(w) - \xi$ then go to Step 2.
        Else Set $\alpha = 0$, return $\alpha$.
Step 2. Set $\beta = \min\{\bar{\alpha}, 2\alpha\}$.
Step 3. If $\alpha = \bar{\alpha}$ or $f(w + \beta p) > f(w) - \xi$, return $\alpha$.
Step 4. Set $\alpha = \beta$ and go to Step 2.
```

The algorithm for bound constrained problems proposed in this section is called DFNDFL because its two main phases are inspired by two previously proposed algorithms. These two algorithms are named, respectively, DFN (an algorithm for continuous nonsmooth optimization, see [21]) and DFLINT (an algorithm for mixed-integer optimization, see [35]). They can be both freely downloaded from the DFL library at http://www.iasi.cnr.it/~liuzzi/DFL. DFNDFL performs an alternate minimization along continuous and discrete variables and is hence divided into two phases (see Algorithm 3 for a detailed scheme). Starting from a point $x_0 \in X \cap \mathcal{Z}$, in Phase 1 the minimization over the continuous variables is performed by using the Projected Continuous Search (**Step 7**). If the line search fails, i.e., $\alpha_k^c = 0$, the tentative stepsize for continuous search directions is reduced (**Step 9**), otherwise the current iterate is updated (**Step 11**). Then, in Phase 2.A, the directions in the set $D \subset D^z(\tilde{x}_k)$, where $\tilde{x}_k$ is the current iterate obtained at the end of Phase 1, are investigated through the Discrete Search (**Step 16**). If the stepsize returned by the line search performed along a given primitive direction is 0, the corresponding tentative stepsize is halved (**Step 18**), otherwise the current iterate is updated (**Step 20**). The directions in D are explored until either a point leading to a sufficient decrease in the objective function is found or D contains no direction to explore. Note that the strategy starts with a subset of $D^z(x_0)$, namely D, and gradually adds directions to it (see Phase 2.B) throughout the iterations. This choice enables the algorithm to reduce the computational cost. If $\tilde{x}_k$ obtained at the end of Phase 1 is not modified in Phase 2.A or a direction in D_k along which the Discrete Search does not fail with $\tilde{\alpha}_k^{(d)} = 1$ exists, D_{k+1} is set equal to D_k and D is set equal to D_{k+1} (**Step 32**). Otherwise, ξ_k is reduced (**Step 24**) and, if all the feasible primitive discrete directions at $\tilde{x}_k$ have been generated, D_{k+1} and D are not changed compared to the previous iteration (**Step 26**). Instead, when $D_k \subset D^z(\tilde{x}_k)$, D_k is enriched with new feasible primitive discrete directions (**Steps 28–29**) and the initial tentative stepsizes of the new directions are set equal to 1.

It is worth noticing that the positive parameter ξ_k plays an important role in the algorithm, since it rules the sufficient decrease of the objective function value within the Discrete Search. The update of the parameter is performed at **Step 24**. More specifically, we shrink the value of the parameter when both the current iterate is not updated in Phase 2.A and the Discrete Search fails (with $\tilde{\alpha}_k^{(d)} = 1$) along each direction in D_k. Hence, Algorithm DFNDFL is not the mere union of the two algorithms DFN [21], for continuous nonsmooth problems, and DFLINT [35], for integer problems. In order to prove convergence of the proposed algorithm to stationary points, optimization with respect to the discrete variables (i.e., phase 2.A of DFNDFL) must be indeed carried out by guaranteeing sufficient decrease with respect to the parameter ξ_k. Such a parameter is possibly decreased in phase 2.B of the algorithm (when the whole iteration "fails") and eventually goes to zero.

The following propositions guarantee that the algorithm is well-defined. In particular, Proposition 1 follows the same reasoning as [21, Proposition 2.4].

Proposition 1 *The Projected Continuous Search cannot infinitely cycle between* **Step 4** *and* **Step 6**.

Proof We assume by contradiction that in the Projected Continuous Search an infinite monotonically increasing sequence of positive numbers $\{\beta_j\}$ exists such that $\beta_j \to \infty$ for $j \to \infty$ and

$$f([w + \beta_j \tilde{p}]_{[l,u]}) \leq f(w) - \gamma \beta_j^2.$$

Since by the instructions of the procedures we have that $[w + \beta_j \tilde{p}]_{[l,u]} \in X \cap \mathcal{Z}$, the previous relation is in contrast with the compactness of X, by definition of compact set, and with the continuity of function f. These arguments conclude the proof. □

Proposition 2 *The Discrete Search cannot infinitely cycle between* **Step 2** *and* **Step 4**.

Algorithm 3 DFNDFL

```
    $\boxed{\textbf{DATA}}$
1: Let $x_0 \in X \cap \mathcal{Z}$, $\xi_0 > 0$, $\theta \in (0,1)$;
2: let $\{s_k\}$ be a sequence such that $s_k \in S(0,1)$ for all $k$;
3: let $\tilde{\alpha}_0^c = 1$ be the initial stepsize along $s_k$;
4: let $D = D_0 \subset D^z(x_0)$ be a set of initial feasible primitive discrete directions at point $x_0$;
5: let $\tilde{\alpha}_0^{(d)} = 1$ be the initial stepsizes along $d \in D$.
6: For $k = 0, 1, \dots$
        $\boxed{\textbf{PHASE 1 - Explore continuous variables}}$
7:      Compute $\alpha_k^c$ and $\tilde{s}_k$ by the Projected Continuous Search($\tilde{\alpha}_k^c, x_k, s_k; \alpha_k^c, \tilde{s}_k$).
8:      If ($\alpha_k^c = 0$) then
9:          $\tilde{\alpha}_{k+1}^c = \theta\tilde{\alpha}_k^c$ and $\tilde{x}_k = x_k$,
10:     else
11:         $\tilde{\alpha}_{k+1}^c = \alpha_k^c$ and $\tilde{x}_k = [x_k + \alpha_k^c \tilde{s}_k]_{[l,u]}$.
12:     End If
        $\boxed{\textbf{PHASE 2.A - Explore discrete variables}}$
13:     Set $y^+ = \tilde{x}_k$.
14:     While $D \neq \emptyset$ and $y^+ = \tilde{x}_k$ do
15:         Choose $d \in D$, set $D = D \setminus \{d\}$ and $y = y^+$.
16:         Compute $\alpha$ by the Discrete Search($\tilde{\alpha}_k^{(d)}, y, d, \xi_k; \alpha$).
17:         If $\alpha = 0$ then
18:             Set $y^+ = y$ and $\tilde{\alpha}_{k+1}^{(d)} = \max\{1, \lfloor \tilde{\alpha}_k^{(d)}/2 \rfloor\}$,
19:         else
20:             Set $y^+ = y + \alpha d$ and $\tilde{\alpha}_{k+1}^{(d)} = \alpha$.
21:         End If
22:     End While
        $\boxed{\textbf{PHASE 2.B - Update the set of discrete search directions}}$
23:     If $y^+ = \tilde{x}_k$ and Discrete Search fails with $\tilde{\alpha}_k^{(d)} = 1$ for all $d \in D_k$ then
24:         Set $\xi_{k+1} = \theta\xi_k$.
25:         If $D_k \supseteq D^z(\tilde{x}_k)$ then
26:             Set $D_{k+1} = D_k$ and $D = D_{k+1}$,
27:         else
28:             Generate $D_{k+1}$ such that $D_{k+1} \subseteq D^z(\tilde{x}_k)$ and $D_{k+1} \supset D_k$, set $D = D_{k+1}$.
29:             Set $\tilde{\alpha}_{k+1}^{(d)} = 1$ for all $d \in D_{k+1} \setminus D_k$.
30:         End If
31:     else
32:         Set $D_{k+1} = D_k$ and $D = D_{k+1}$.
33:     End If
        $\boxed{\textbf{PHASE 3 - Update iterates}}$
34:     Find $x_{k+1} \in X \cap \mathcal{Z}$ such that $f(x_{k+1}) \leq f(y^+)$.
35: End For
```

Proof First, note that since $X \cap \mathcal{Z}$ is compact, the maximum stepsize $\bar{\alpha}$ computed at **Step 0** is finite. Let us consider the j-th iteration of the Discrete search, we have $\beta = 2^j \min\{\bar{\alpha}, \tilde{\alpha}\}$. Now, given the termination condition at **Step 3**, index j cannot exceed $\lceil \log(\bar{\alpha}/\min\{\bar{\alpha}, \tilde{\alpha}\}) \rceil$, thus concluding the proof □

The following proposition shows that the Projected Continuous Search returns stepsizes that eventually go to zero. This proposition will be used to prove stationarity with respect to the continuous variables.

Proposition 3 *Let $\{\alpha_k^c\}$ and $\{\tilde{\alpha}_k^c\}$ be the sequences yielded by Algorithm DFNDFL. Then*

$$\lim_{k\to\infty} \max\{\alpha_k^c, \tilde{\alpha}_k^c\} = 0. \tag{3.1}$$

Proof The proof follows with minor modifications from the proof of Proposition 2.5 in [21] by considering that also here inequality (2.5) of [21] holds. □

The following proposition will be used to show that every limit point of a particular subsequence of iterates is a local minimum with respect to the discrete variables.

Proposition 4 *Let $\{\xi_k\}$ be the sequence produced by Algorithm DFNDFL. Then*

$$\lim_{k\to\infty} \xi_k = 0.$$

Proof By the instruction of Algorithm DFNDFL, it follows that $0 < \xi_{k+1} \le \xi_k$ for all k, meaning that the sequence $\{\xi_k\}$ is monotonically nonincreasing. Hence, $\{\xi_k\}$ converges to a limit $M \ge 0$. Suppose, by contradiction, that $M > 0$. This implies that an index $\bar{k} > 0$ exists such that

$$\xi_{k+1} = \xi_k = M \tag{3.2}$$

for all $k \ge \bar{k}$. Moreover, for every index $k \ge \bar{k}$, a direction $d \in D^z(\tilde{x}_k)$ exists such that

$$f(x_{k+1}) \le f(\tilde{x}_k + \alpha_k^{(d)} d) \le f(\tilde{x}_k) - \xi_k = f(\tilde{x}_k) - M \le f(x_k) - M, \tag{3.3}$$

where the inequalities follow from the instructions of Algorithms 2–3 and the equality follows from (3.2). Relation (3.3) implies $f(x_k) \to -\infty$, which is in contrast with the assumption that f is continuous on the compact set X. This concludes the proof. □

Remark 2 By the preceding proposition and the updating rule of the parameter ξ_k used in Algorithm DFNDFL, it follows that the set

$$H = \{k : \xi_{k+1} < \xi_k\}$$

is infinite.

The previous result is used to prove the next lemma, which in turn is essential to prove the global convergence result related to the continuous variables. This lemma states that the asymptotic convergence properties of the sequence $\{s_k\}$ still hold when the projection operator is adopted. Its proof closely resembles the proof in [21, Lemma 2.6].

Lemma 1 *Let $\{x_k\}$ and $\{s_k\}$ be the sequence of points and the sequence of continuous search directions produced by Algorithm DFNDFL, respectively, and $\{\eta_k\}$ be a sequence such that $\eta_k > 0$, for all k. Further, let K be a subset of indices such that*

$$\lim_{k\to\infty, k\in K} x_k = \bar{x}, \tag{3.4}$$

$$\lim_{k\to\infty, k\in K} s_k = \bar{s}, \tag{3.5}$$

$$\lim_{k\to\infty, k\in K} \eta_k = 0. \tag{3.6}$$

with $\bar{x} \in X \cap \mathcal{Z}$ and $\bar{s} \in D^c(\bar{x})$, $\bar{s} \neq 0$. Then,

(i) *for all $k \in K$ sufficiently large,*

$$[x_k + \eta_k s_k]_{[l,u]} \neq x_k,$$

(ii) *the following limit holds*

$$\lim_{k\to\infty, k\in K} v_k = \bar{s},$$

where

$$v_k = \frac{[x_k + \eta_k s_k]_{[l,u]} - x_k}{\eta_k}. \tag{3.7}$$

Proof Since $x_k \in X \cap \mathcal{Z}$ and (3.4) holds, we have necessarily that $(x_k)_z = \bar{x}_z$ for $k \in K$ sufficiently large. Now, the proof follows with minor modifications from the proof of Lemma 2.6 in [21]. □

The convergence result related to the continuous variables is proved in the next proposition. It will be used in the main convergence result at the end of the section. It basically states that every limit point of the subsequence of iterates defined by the set H (see Remark 2) is a stationary point with respect to the continuous variables.

Proposition 5 *Let $\{x_k\}$ be the sequence of points produced by Algorithm DFNDFL. Let $H \subseteq \{1, 2, \dots\}$ be defined as in Remark 2 and let $\bar{x} \in X \cap \mathcal{Z}$ be any accumulation* *point of* $\{x_k\}_H$*. If the subsequence* $\{s_k\}_H$*, with* $(s_k)_i = 0$ *for* $i \in I^z$*, is dense in the unit sphere (see Definition 2), then* $\bar{x}$ *satisfies*

$$f_c^\circ(\bar{x};s) \geq 0 \qquad \text{for all } s \in D^c(\bar{x}). \tag{3.8}$$

Proof For any accumulation point $\bar{x}$ of $\{x_k\}_H$, let $K \subseteq H$ be an index set such that

$$\lim_{k\to\infty, k\in K} x_k = \bar{x}. \tag{3.9}$$

Notice that, for all $k \in K$, $(\tilde{x}_k)_z = (x_k)_z$ and $\tilde{\alpha}_k^{(d)} = 1$, $d \in D_k$, by the instructions of Algorithm DFNDFL. Hence, for all $k \in K$, by recalling (3.9), the discrete variables are no longer updated.

Now, recalling Proposition 3 and Lemma 1, and repeating the proof of [21, Proposition 2.7], it can be shown that no direction $\bar{s} \in D^c(\bar{x}) \cap S(0, 1)$ can exist such that

$$f_c^\circ(\bar{x};\bar{s}) < 0, \tag{3.10}$$

thus concluding the proof. □

The next proposition states that every limit point of the subsequence of iterates defined by the set H (see Remark 2) is a local minimum with respect to the discrete variables. It will be used in the main convergence result at the end of the section.

Proposition 6 *Let* $\{x_k\}$, $\{\tilde{x}_k\}$, *and* $\{\xi_k\}$ *be the sequences produced by Algorithm DFNDFL. Let* $H \subseteq \{1, 2, \dots\}$ *be defined as in Remark 2 and* $x^* \in X \cap \mathcal{Z}$ *be any accumulation point of* $\{x_k\}_H$*, then*

$$f(x^*) \leq f(\bar{x}), \qquad \text{for all } \bar{x} \in \mathcal{B}^z(x^*).$$

Proof Let $K \subseteq H$ be an index set such that

$$\lim_{k\to\infty, k\in K} x_k = x^*.$$

For every $k \in K \subseteq H$, we have

$$\begin{aligned} (\tilde{x}_k)_z &= (x_k)_z, \\ \tilde{\alpha}_k^{(d)} &= 1, \quad d \in D_k, \end{aligned}$$

meaning that the discrete variables are no longer updated by the Discrete Search.

Let us consider any point $\bar{x} \in \mathcal{B}^z(x^*)$. By the definition of discrete neighborhood $\mathcal{B}^z(x^*)$, a direction $\bar{d} \in D^z(x^*)$ exists such that

$$\bar{x} = x^* + \bar{d}. \tag{3.11}$$

Recalling the steps in Algorithm DFNDFL, we have, for all $k \in H$ sufficiently large, that

$$(x^*)_z = (x_k)_z = (\tilde{x}_k)_z. \tag{3.12}$$

Further, by Proposition 3, we have

$$\lim_{k\to\infty, k\in K} \tilde{x}_k = x^*.$$

Then, for all $k \in K$ sufficiently large, (3.11) and (3.12) imply

$$(x_k + \bar{d})_z = (\tilde{x}_k + \bar{d})_z = (x^* + \bar{d})_z = (\bar{x})_z.$$

Hence, for all $k \in K$ sufficiently large, by the definition of discrete neighborhood we have $\bar{d} \in D^z(\tilde{x}_k)$ and

$$\tilde{x}_k + \bar{d} \in X \cap \mathcal{Z}.$$

Then, since $k \in K \subseteq H$, by the definition of H we have

$$f(\tilde{x}_k + \bar{d}) > f(\tilde{x}_k) - \xi_k. \tag{3.13}$$

Now, by Proposition 4, and taking the limit for $k \to \infty$, with $k \in K$, in (3.13), the result follows. □

Now, the main convergence result of the algorithm can be proved.

Theorem 1 *Let $\{x_k\}$ be the sequence of points generated by Algorithm DFNDFL. Let $H \subseteq \{1, 2, \dots\}$ be defined as in Remark 2 and let $\{s_k\}_H$, with $(s_k)_i = 0$ for $i \in I^z$, be a dense subsequence in the unit sphere. Then,*

(i) *a limit point of $\{x_k\}_H$ exists;*
(ii) *every limit point x^* of $\{x_k\}_H$ is stationary for Problem (2.2).*

Proof As regards point (i), since $\{x_k\}_H$ belongs to the compact set $X \cap \mathcal{Z}$, it admits limit points. The prove of point (ii) follows by considering Propositions 5 and 6 .

□

4 An algorithm for nonsmooth nonlinearly constrained problems

In this section, the nonsmooth nonlinearly constrained problem defined in Problem (1.3) is considered. The nonlinear constraints are handled through a simple penalty approach (see, e.g., [21]). In particular, given a positive parameter $\varepsilon > 0$, the following penalty function is introduced

$$P(x;\varepsilon) = f(x) + \frac{1}{\varepsilon}\sum_{i=1}^{m} \max\left\{0, g_i(x)\right\},$$

which allows to define the following bound constrained problem

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \ & P(x;\varepsilon) \\ \text{s.t.} \ & x \in X \cap \mathcal{Z}. \end{aligned} \tag{4.1}$$

Hence, only the nonlinear constraints are penalized and the minimization is performed over the set $X \cap \mathcal{Z}$. The algorithm described in Sect. 3 is thus suited for solving this problem, as highlighted in the following remark.

Remark 3 Observe that, for any $\varepsilon > 0$, the structure and properties of Problem (4.1) are the same as Problem (2.2). The Lipschiz continuity with respect to the continuous variables of the penalty function $P(x;\varepsilon)$ follows by the Lipschitz continuity of f and g_i, with $i \in \{1,\dots,m\}$. In particular, called L_f and L_{g_i} the Lipschitz constants of f and g_i, respectively, we have that the Lipschitz constant of the penalty function $P(x;\varepsilon)$ is

$$L \leq L_f + \frac{1}{\varepsilon}\sum_{i=1}^{m} L_{g_i}.$$

To prove the equivalence between Problem (1.3) and Problem (4.1), we report an extended version of the Mangasarian-Fromowitz Constraint Qualification (EMFCQ) [24, 39] for Problem (1.3), which takes into account its mixed-integer structure. This condition states that at a point that is infeasible for Problem (1.3), a direction feasible with respect to $X \cap \mathcal{Z}$ (according to Definitions 4 and 8) that guarantees a reduction in the constraint violation exists.

Assumption 7 (EMFCQ for mixed-integer problems) Let us consider Problem (1.3). For any $x \in (X \cap \mathcal{Z}) \setminus \mathcal{F}$, one of the following conditions holds:

(i) a direction $s \in D^c(x)$ exists such that

$$(\xi^{g_i})^\top s < 0,$$

for all $\xi^{g_i} \in \partial_c g_i(x)$ with $i \in \{h \in \{1,2,\dots,m\} \ : \ g_h(x) \geq 0\}$;

(ii) a direction $\bar{d} \in D^z(x)$ exists such that

$$\sum_{i=1}^{m} \max\{0, g_i(x+\bar{d})\} < \sum_{i=1}^{m} \max\{0, g_i(x)\}.$$

In order to prove the main convergence properties of the algorithm in this case, the equivalence between the original constrained Problem (1.3) and the penalized Problem (4.1) must be established first. The proof of this result is very technical and quite similar to analogous results from [21]. We report it in the Appendix for the sake of major clarity.

Exploiting this technical result, the algorithm proposed in Sect. 3 can be used to solve Problem (4.1), provided that the penalty parameter is sufficiently small, as stated in the next proposition. The algorithmic scheme designed for solving Problem (1.3) is obtained from Algorithm DFNDFL by replacing $f(x)$ with $P(x;\varepsilon)$, where

$\varepsilon > 0$ is a sufficiently small value. We point out that in this new scheme both the linesearch procedures are performed by replacing $f(x)$ with $P(x;\varepsilon)$ as well. We refer to this new scheme as DFNDFL–CON.

Proposition 8 *Let Assumption 7 hold and let $\{x_k\}$ be the sequence produced by Algorithm DFNDFL–CON. Let $H \subseteq \{1, 2, \dots\}$ be defined as in Remark 2 and let $\{s_k\}_H$, with $(s_k)_i = 0$for $i \in I^z$, be a dense subsequence in the unit sphere. Then, $\{x_k\}_H$ admits limit points. Furthermore, a threshold value ε^* exists such that for all $\varepsilon \in (0, \varepsilon^*]$ every limit point x^* of $\{x_k\}_H$ is stationary for Problem (1.3).*

Proof The proof follows from Proposition 14 and Theorem 1. □

5 Numerical experiments

In this section, results of the numerical experiments performed on a set of test problems selected from the literature are reported. In particular, state-of-the-art solvers are used as benchmarks to test the efficiency and reliability of the proposed algorithm. First, the bound constrained case is considered, then nonlinearly constrained problems are tackled. In both cases, to improve the performance of DFNDFL, a modification to Phase 1 is introduced by drawing inspiration from the algorithm CS-DFN proposed in [21] for continuous nonsmooth problems. In particular, recalling that $I^c \cup I^z = \{1, 2, \dots, n\}$, the change consists in investigating the set of coordinate directions $\{\pm e^1, \pm e^2 \dots, \pm e^{|I^c|}\}$ before exploring a direction from the sequence $\{s_k\}$. Since this set is constant over the iterations, the actual stepsizes $\alpha_k^{(i)}$ and tentative stepsizes $\tilde{\alpha}_k^{(i)}$ can be stored for each coordinate direction i, with $i \in \{1, 2, \dots, n\}$. These stepsizes are reduced whenever the projected continuous line search, i.e., Algorithm 1, does not determine any point that satisfies the sufficient decrease condition. When their values become sufficiently small, a direction from the dense sequence s_k is explored. This improvement allows the algorithm to benefit from the presence of the stepsizes $\alpha_k^{(i)}$ and $\tilde{\alpha}_k^{(i)}$, whose values depend on the knowledge across the iterations of the sensitivity of the objective function over the coordinate directions. The use of those coordinate directions hence allows to somehow capture the local behaviour of the function through the actual/tentative stepsizes. So, we can take advantage of the information gathered in previous iterations through those stepsizes when searching for a new point. Furthermore, we can use the dense set of directions only when really needed (i.e., only when approaching a point where the function is actually nonsmooth). Therefore, the efficiency of the modified DFNDFL is expected to be higher.

The codes related to the DFNDFL and DFNDFL–CON algorithms, together with the test problems used in the experiments are freely available for download at the *Derivative-Free Library* web page http://www.iasi.cnr.it/~liuzzi/DFL/.

5.1 Algorithms for benchmarking

The algorithms selected as benchmarks are listed below:

- DFL *box* (see [33]), a derivative-free linesearch algorithm for bound constrained problems.
- DFL *gen* (see [34]), a derivative-free linesearch algorithm for nonlinearly constrained problems.
- RBFOpt (see [16]), an open-source library RBFOpt for solving black-box bound constrained optimization problems with expensive function evaluations.
- NOMAD v.3.9.1 (see [9]), a software package which implements the MADS algorithm.
- MISO (Mixed-Integer Surrogate Optimization) framework, a model-based approach using surrogates [41].

All the algorithms reported above support mixed-integer problems, thus being suited for the comparison with the algorithm proposed in this work. The maximum allowable number of function evaluations in each experiment is 5000. All the codes have been run on an Intel Core i7 10th generation CPU PC running Windows 10 with 16GB of memory. More precisely, all the test problems, DFNDFL, DFL *box*, DFL *gen* and RBFOpt are coded in python and have been run using python 3.8. NOMAD, on the other hand, is delivered as a collection of C++ codes and has been run using the provided PyNomad python interface. As for MISO, it is coded in matlab and has been run using Matlab R2020b but using the python coded problems through the matlab python engine.

As regards the parameters used in both DFNDFL and DFL, the values used in the experiments are $\gamma = 10^{-6}$, $\delta = 0.5$, $\xi_0 = 1$, and $\theta = 0.5$. Moreover, the initial tentative stepsizes along the coordinate directions $\pm e^i$ and s_k of the modified DFNDFL are

$$\begin{aligned}\tilde{\alpha}_0^i &= (u^i - \ell^i)/2 \qquad \text{for all } i \in I^c,\\ \tilde{\alpha}_0 &= \frac{1}{n}\sum_{i=1}^{n} \tilde{\alpha}_0^i,\end{aligned}$$

while for the discrete directions d the initial tentative stepsize $\tilde{\alpha}_0^{(d)}$ is fixed to 1.

Another computational aspect that needs to be further discussed is the generation of the continuous and discrete directions. Indeed, in Phases 1 and 2 of DFNDFL, new search directions might be generated to thoroughly explore neighborhoods of the current iterate. To this end, a dense sequence of directions $\{s_k\}$ is required in Phase 1 to explore the continuous variables and, in particular, the Sobol sequence [13, 48] is used. Similarly, in Phase 2, new primitive discrete directions must be generated when some suitable conditions hold. In these cases, the Halton sequence [26] is used.

As concerns the parameters used for running RBFOpt and NOMAD, while the former is executed by using the default values, for the latter two different algorithms are considered. The first one is based on the default settings, while the second one results from disabling the usage of models in the search phase, which precisely is obtained by setting `DISABLE MODELS`. This second version is denoted in the remainder of this document as NOMAD (w/o mod).

5.2 Test problems

The comparison between Algorithm DFNDFL and some state-of-the-art solvers is reported for 24 bound constrained problems. The first 16 problems, which are related to minimax and nonsmooth unconstrained optimization problems, have been selected from [38, Sections 2 and 3], while the remaining 8 problems have been chosen from [41, 42]. The problems are listed in Table 1 along with the respective number of continuous (n_c) and discrete (n_z) variables.

Since the problems from [38] are unconstrained, in order to suit the class of problems addressed in this work, the following bound constraints are considered for each variable

$$\ell^i = (\tilde{x}_0)^i - 10 \le \tilde{x}^i \le (\tilde{x}_0)^i + 10 = u^i \quad \text{for all } i \in \{1, 2, \dots, n\},$$

where $\tilde{x}_0$ is the starting point. Furthermore, since the problems in [38] have only continuous variables, the rule applied to obtain mixed-integer problems is to consider a number of integer variables equal to $n_z = \lfloor n/2 \rfloor$ and a number of continuous variables equal to $n_c = \lceil n/2 \rceil$, where n denotes the dimension of each original problem and $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ are the floor and ceil operators, respectively.

More specifically, let us consider both the continuous bound constrained optimization problems from [38], whose formulation is

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \ & \tilde{f}(\tilde{x}) \\ \text{s.t. } & \ell^i \le \tilde{x}^i \le u^i \quad \text{for all } i \in \{1, 2, \dots, n\}, \\ & \tilde{x}_i \in \mathbb{R} \quad\quad\quad\ \ \text{for all } i \in \{1, 2, \dots, n\}, \end{aligned} \tag{5.1}$$

and the original mixed-integer bound constrained problems from [41, 42], which can be stated as

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \ & \tilde{f}(\tilde{x}) \\ \text{s.t. } & \ell^i \le \tilde{x}^i \le u^i \quad \text{for all } i \in \{1, 2, \dots, n\}, \\ & x_i \in \mathbb{R} \quad\quad\quad\ \ \text{for all } i \in I^c, \\ & x_i \in \mathbb{Z} \quad\quad\quad\ \ \text{for all } i \in I^z. \end{aligned} \tag{5.2}$$

The resulting mixed-integer problem we deal with in here can be formulated as follows

Table 1 Bound constrained test problems collection

Problem name	Source	n_c	n_z	Problem name	Source	n_c	n_z
gill	[38]	5	5	goffin	[38]	25	25
l1hilb(20)	[38]	10	10	l1hilb(30)	[38]	15	15
l1hilb(40)	[38]	20	20	l1hilb(50)	[38]	25	25
maxl	[38]	10	10	maxq(20)	[38]	10	10
maxq(30)	[38]	15	15	maxq(40)	[38]	20	20
maxq(50)	[38]	25	25	maxquad	[38]	5	5
mxhilb	[38]	25	25	osborne 2	[38]	6	5
polak 2	[38]	5	5	polak 3	[38]	6	5
shell dual	[38]	8	7	steiner 2	[38]	6	6
tr48	[38]	24	24	watson	[38]	10	10
wong2	[38]	5	5	wong3	[38]	10	10
SO-I prob. 7	[42]	5	5	SO-I prob. 9	[42]	6	6
SO-I prob.10	[42]	15	15	SO-I prob.13	[42]	5	5
SO-I prob.15	[42]	6	6	MISO prob. 6	[41]	8	7
MISO prob. 8	[41]	5	10	MISO prob.10	[41]	30	30

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} f(x) \\ &\text{s.t. } \ell^i \le x^i \le u^i \quad \text{for all } i \in I^c, \\ &\quad 0 \le x^i \le 100 \quad \text{for all } i \in I^z, \\ &\quad x_i \in \mathbb{R} \qquad\quad \text{for all } i \in I^c, \\ &\quad x_i \in \mathbb{Z} \qquad\quad \text{for all } i \in I^z, \end{aligned} \tag{5.3}$$

where $f(x) = \tilde{f}(\tilde{x})$ with

$$\tilde{x}^i = \begin{cases} x^i, & \text{for all } i \in I^c, \\ \ell^i + x^i(u^i - \ell^i)/100, & \text{for all } i \in I^z. \end{cases}$$

Moreover, the starting point x_0 adopted for Problem (5.3) is

$$(x_0)^i = \begin{cases} (u^i - \ell^i)/2 & \text{for all } i \in I^c, \\ 50 & \text{for all } i \in I^z. \end{cases}$$

As concerns constrained mixed-integer optimization problems, the performances of Algorithm DFNDFL–CON are assessed on 204 problems with general nonlinear constraints. Such problems are obtained by adding to the 34 bound-constrained problems defined by Problem (5.3) the 6 classes of constraints reported below and proposed in [29]

$$
\begin{array}{ll}
g_j(x) = (3 - 2x_{j+1})x_{j+1} - x_j - 2x_{j+2} + 1 \le 0, & \text{for all } j \in \{1, 2, \dots, n-2\} \\
g_j(x) = (3 - 2x_{j+1})x_{j+1} - x_j - 2x_{j+2} + 2.5 \le 0 & \text{for all } j \in \{1, 2, \dots, n-2\} \\
g_j(x) = x_j^2 + x_{j+1}^2 + x_j x_{j+1} - 2x_j - 2x_{j+1} + 1 \le 0 & \text{for all } j \in \{1, 2, \dots, n-1\} \\
g_j(x) = x_j^2 + x_{j+1}^2 + x_j x_{j+1} - 1 \le 0 & \text{for all } j \in \{1, 2, \dots, n-1\} \\
g_j(x) = (3 - 0.5x_{j+1})x_{j+1} - x_j - 2x_{j+2} + 1 \le 0 & \text{for all } j \in \{1, 2, \dots, n-2\} \\
g_1(x) = \sum_{j=1}^{n-2} ((3 - 0.5x_{j+1})x_{j+1} - x_j - 2x_{j+2} + 1) \le 0 &
\end{array}
$$

Thus, the number of general nonlinear constraints ranges from 1 to 59.

5.3 Data and performance profiles

The comparison among the algorithms is carried out by using data and performance profiles, which are benchmarking tools widely used in derivative-free optimization (see [20, 40]). In particular, given a set S of algorithms, a set P of problems, and a convergence test, data and performance profiles provide complementary information to assess the relative performance among the different algorithms in S when applied to solve problems in P. Specifically, data profiles allow gaining insight on the percentage of problems that are solved (according to the convergence test defined below) by each algorithm within a given budget of function evaluations. On the other hand, performance profiles allow assessing how well an algorithm performs with respect to the others. For each $s \in S$ and $p \in P$, the number of function evaluations required by algorithm s to satisfy the convergence condition on problem p is denoted as $t_{p,s}$. Given a tolerance $0 < \tau < 1$ the convergence test is

$$f(x_k) \le f_L + \tau(\hat{f}(x_0) - f_L),$$

where:

- $f(x_k)$ is the objective function value computed at x_k. When dealing with problems with general constraints, we set to $+\infty$ the value of the objective function at infeasible points;
- $\hat{f}(x_0)$ is the objective function value of the worst feasible point determined by all the solvers (note that in the bound-constrained case, $\hat{f}(x_0) = f(x_0)$);
- f_L is the smallest feasible objective function value computed by any algorithm on the considered problem within the given number of 5000 function evaluations.

The above convergence test requires the best point to achieve a sufficient reduction from the value $\hat{f}(x_0)$ of the objective function at the starting point. Note that the smaller the value of the tolerance τ is, the higher accuracy is required at the best point. In particular, three levels of accuracy are considered in this paper for the parameter τ, namely, $\tau \in \{10^{-1}, 10^{-3}, 10^{-5}\}$.

Performance and data profiles of solver s can be formally defined as follows

$$\rho_s(\alpha) = \frac{1}{|P|}\left|\left\{p \in P : \frac{t_{p,s}}{\min\{t_{p,s'} : s' \in S\}} \le \alpha\right\}\right|,$$
$$d_s(\kappa) = \frac{1}{|P|}\left|\left\{p \in P : t_{p,s} \le \kappa(n_p + 1)\right\}\right|,$$

where n_p is the dimension of problem p. While α indicates that the number of function evaluations required by algorithm s to achieve the best solution is α–times the number of function evaluations needed by the best algorithm, κ denotes the number of simplex gradient estimates, with $n_p + 1$ being the number associated with one simplex gradient. Important features for the comparison are $\rho_s(1)$, which is a measure of the efficiency of the algorithm, since it is the percentage of problems for which the algorithm s performs the best, and the height reached by each profile as the value of α or κ increases, which measures the reliability of the solver.

5.3.1 The bound constrained case

Figure 1 reports performance and data profiles related to the comparison of the algorithms that do not employ models, namely DFNDFL, DFL *box* and NOMAD (without models). In this case, DFNDFL turns out to be the most efficient and reliable algorithm, regardless of the accuracy required. DFL *box* is more efficient than NOMAD for all values of τ, whereas NOMAD is (slightly) more reliable for $\tau = 10^{-1}$ and 10^{-3}. It is worth noticing that the remarkable percentage of problems solved for $\alpha = 1$ is an important result for DFNDFL since it shows that using more sophisticated directions than DFL *box* does not lead to a significant loss of efficiency. We also point out that the initial continuous and primitive search directions used by DFNDFL are the coordinate directions, which are the same as the ones employed in DFL *box*, thus leading to the same behavior of the algorithms in the first iterations. For each value of τ, despite the remarkable efficiency, DFL *box* does not show a strong reliability, which is significantly improved by DFNDFL.

Next we compare DFNDFL against solvers that make use of sophisticated models to improve the search. In particular, we considered NOMAD (using models), RBFOpt and MISO. We point out that these three solvers exploit different kinds of models. In particular, NOMAD takes advantage of quadratic models whereas RBFOpt and MISO make use of radial basis function models. It is important to highlight that both MISO and RBFOpt are designed for a low budget of function evaluations, (i.e., to obtain large improvements at the very beginning of the search); they do not have the capability of finding very accurate local solutions, thus they are in general not expected to be competitive for high precision.

Figure 2 reports performance and data profiles for the three considered levels of accuracy when DFNDFL is compared with the algorithms that make use of models.

From the performance and data profiles, we can note that DFNDFL is competitive with the other methods for low precisions, and gives better results than the other methods, both in terms of efficiency and reliability, when precision gets higher.

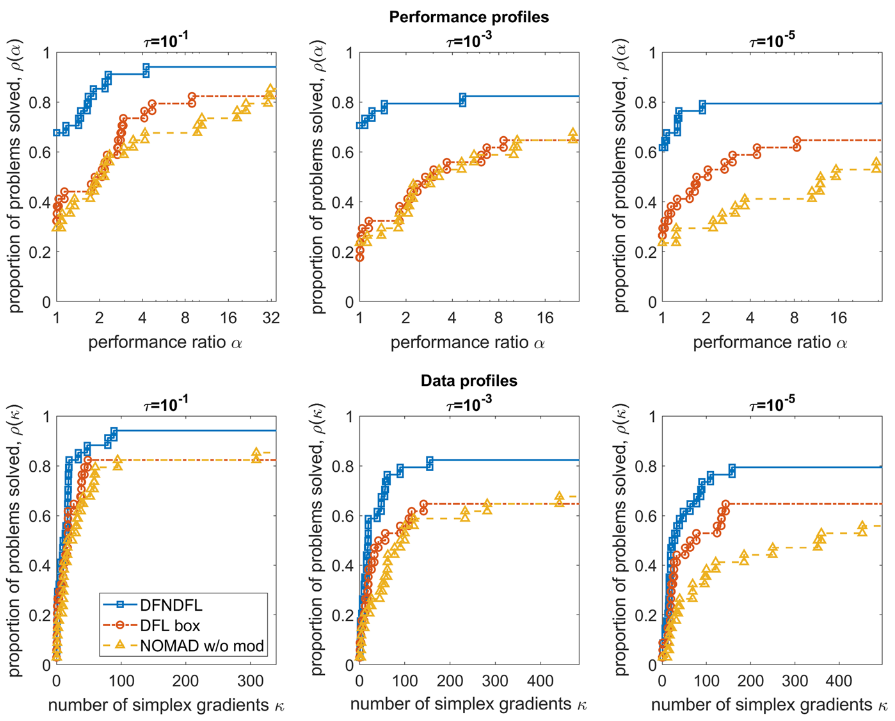

Fig. 1 Performance and data profiles for the comparison among DFNDFL, DFL *box*, NOMAD (not using models) on the 34 bound constrained problems

These numerical results highlight that DFNDFL has a remarkable efficiency and compares favorably to the state-of-the-art-solvers in terms of reliability, thus confirming and strengthening the properties of DFL *box* and providing a noticeable contribution to the derivative-free optimization solvers for bound constrained problems.

5.3.2 The nonlinearly constrained case

The algorithms adopted for comparison in this case are DFL *gen* and the two versions of NOMAD (w/ and w/o models). We point out that the handling of the constraints in NOMAD is performed by the progressive/extreme barrier approach (see [7, 10, 32]) by specifying the PEB constraints type. We would like to highlight that we only used NOMAD and the constrained version of DFL *box* (namely DFL *gen*) in this further comparison, due to their better performances in the bound constrained case and the explicit handling of nonlinearly constrained problems.

Figure 3 reports performance and data profiles, for the comparison of DFNDFL–CON, DFL *gen* and NOMAD (not using models). The figure quite clearly shows that DFNDFL–CON is the most efficient and reliable algorithm, and the difference with the other algorithms significantly grows as the level of accuracy increases. It is important to highlight that using the primitive directions allows our algorithm to

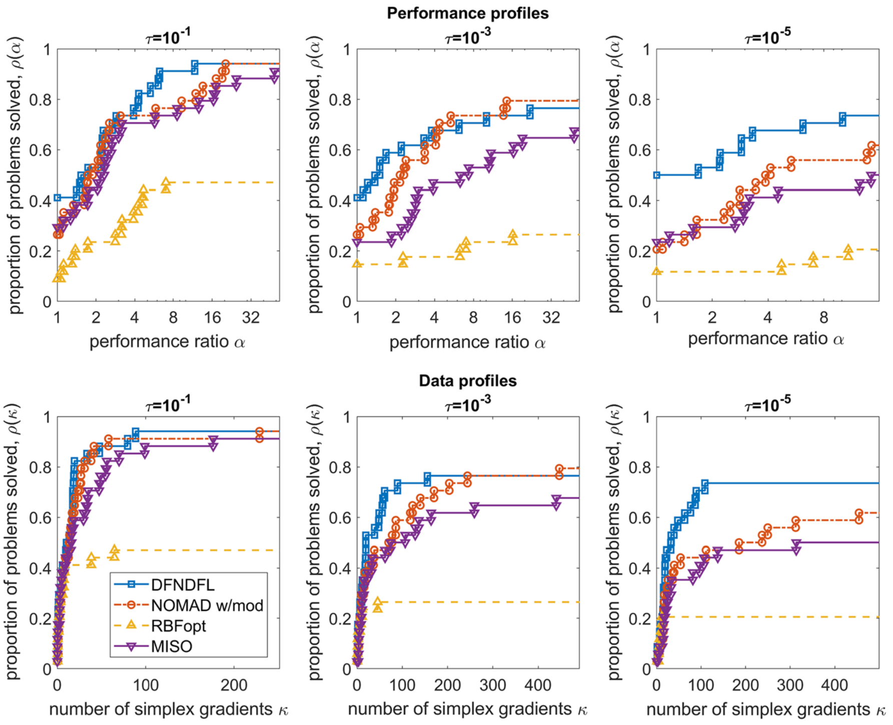

Fig. 2 Performance and data profiles for the comparison among DFNDFL, NOMAD (using models), RBFOpt and MISO on the 34 bound constrained problems

improve the strategy of DFL, which only uses the set of coordinate directions. This results in a larger percentage of problems solved by DFNDFL–CON, even when compared with NOMAD (w/ mod).

Finally, Fig. 4 reports the comparison between DFNDFL–CON and NOMAD using models on the set of 204 constrained problems. Also in this case, it emerges that DFNDFL–CON is competitive with NOMAD both in terms of data and performance profiles.

To conclude, these numerical results show that DFNDFL–CON has remarkable efficiency and reliability when compared to state-of-the-art-solvers.

6 Conclusions

In this paper, new linesearch-based methods for mixed-integer nonsmooth optimization problems have been developed assuming that first-order information on the problem functions is not available. First, a general framework for bound

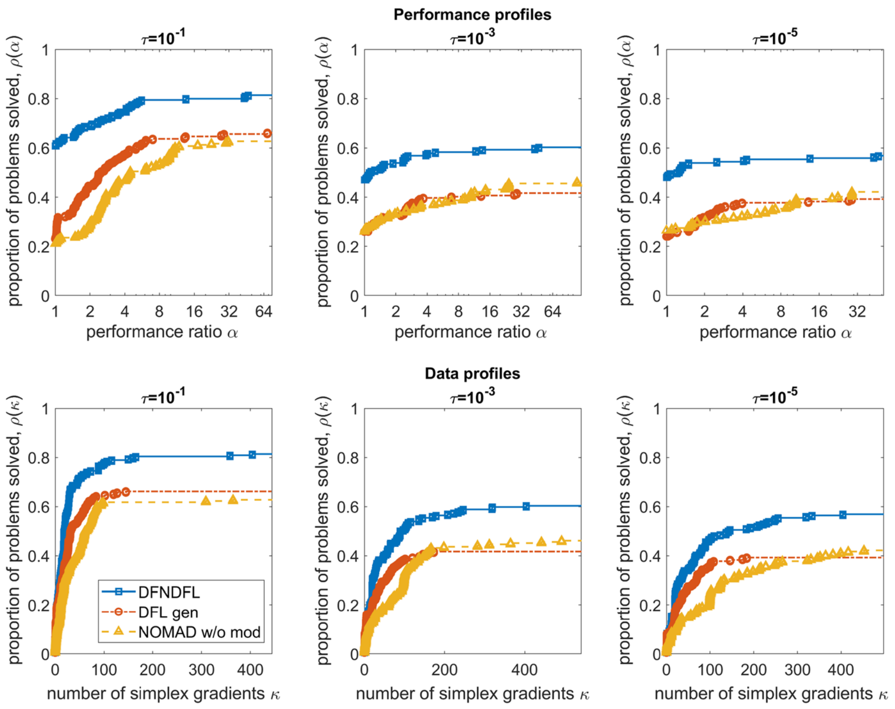

Fig. 3 Performance and data profiles for the comparison among DFNDFL–CON, DFL gen, NOMAD (w/o models) on the 204 nonlinearly constrained problems

constrained problems has been described. Then, an exact penalty approach has been proposed and embedded into the framework for the bound constrained case, thus allowing to tackle the presence of nonlinear (possibly nonsmooth) constraints. Two different sets of directions are adopted to deal with the mixed variables. On the one hand, a dense sequence of continuous search directions is required to detect descent directions. On the other hand, primitive directions are employed to suitably explore the integer lattice thus avoiding to get trapped into bad points.

Numerical experiments have been performed on both bound and nonlinearly constrained problems. The results highlight that the proposed algorithms have good performances when compared with some state-of-the-art-solvers, thus providing a good tool for handling the considered class of derivative-free optimization problems.

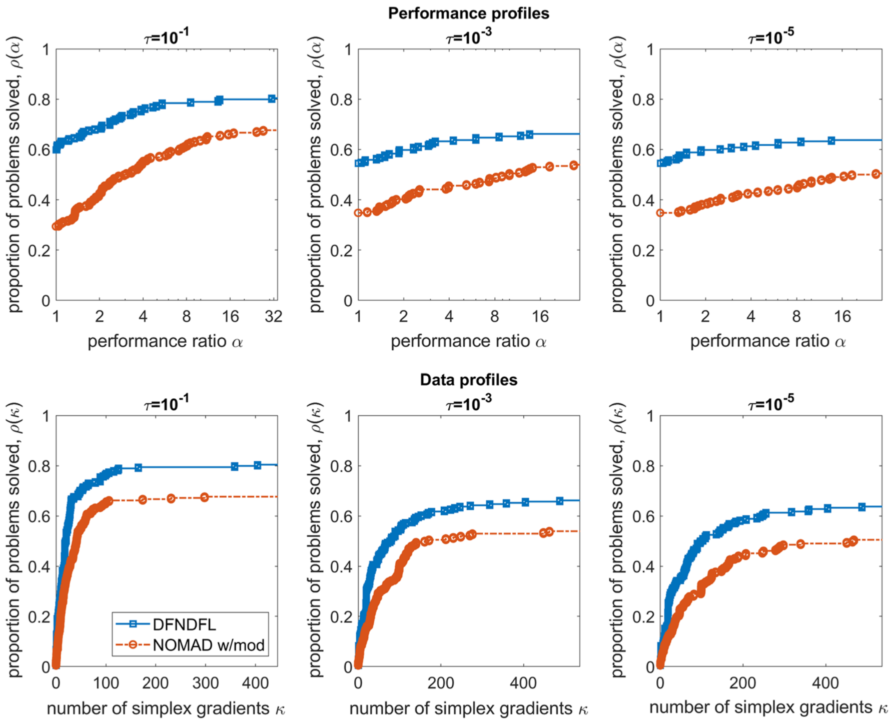

Fig. 4 Performance and data profiles for the comparison between DFNDFL–CON, and NOMAD (w/ models) on the 204 nonlinearly constrained problems

Appendix

The following result simply establishes a connection between minimum points, stationary points and Clarke stationary points of Problem (2.2). It will be used here to relate Problem (1.3) and Problem (4.1).

Lemma 2 *Any local minimum point of Problem* (2.2) *is a stationary point. Furthermore, any stationary point for Problem* (2.2) *is a Clarke stationary point.*

Proof By Definition 7 a local minimum is stationary with respect to the continuous variables. Thus, it is stationary according to Definition 10.

Since, by (2.1) and (2.5), we have that

$$f_c^{Cl}(x^*;s) \geq f_c^{\circ}(x^*;s) \geq 0,$$

it follows that a stationary point is also a Clarke stationary point. □

First of all, we report a technical proposition which will be useful in the following.

Proposition 9 *Let $\{x_k\} \subset X \cap \mathcal{Z}$ for all k, and $\{x_k\} \to \bar{x} \in X \cap \mathcal{Z}$ for $k \to \infty$. Then, for k sufficiently large,*

$$D^c(\bar{x}) \subseteq D^c(x_k).$$

Proof Since $x_k \in X \cap \mathcal{Z}$ and $x_k \to \bar{x}$, we have necessarily that $(x_k)_z = \bar{x}_z$ for k sufficiently large. Furthermore, for k sufficiently large, we also have that the following inclusions hold:

$$\begin{aligned}
N(x_k) &= \{i \in I^c \,:\, l_i < (x_k)_i < u_i\} \supseteq N(\bar{x}) = \{i \in I^c \,:\, l_i < \bar{x}_i < u_i\},\\
L(x_k) &= \{i \in I^c \,:\, (x_k)_i = l_i\} \subseteq L(\bar{x}) = \{i \in I^c \,:\, \bar{x}_i = l_i\},\\
U(x_k) &= \{i \in I^c \,:\, (x_k)_i = u_i\} \subseteq U(\bar{x}) = \{i \in I^c \,:\, \bar{x}_i = u_i\}.
\end{aligned}$$

Now, the result follows by considering $\bar{d} \in D^c(\bar{x})$ and recalling the definition of $D^c(\bar{x})$ (i.e., Definition 8) and the above inclusions. □

We first prove the equivalence between local minimum points and global minimum points of the two problems. Then, we prove that any Clarke stationary point of Problem (4.1) (according to Definition 11) is a stationary point for Problem (1.3) (according to Definition 13).

Proposition 10 *Let Assumption 7 hold. A threshold value $\varepsilon^* > 0$ exists such that the function $P(x;\varepsilon)$ has no Clarke stationary points in $(X \cap \mathcal{Z}) \setminus \mathcal{F}$ for any $\varepsilon \in (0, \varepsilon^*]$.*

Proof The proof of this proposition is very similar to the proof of [21, Proposition B.1]. However, we report it since the presence of discrete variables slightly changes the reasoning. By contradiction, we assume that for any integer k, an $\varepsilon_k \le 1/k$ and a stationary point for Problem (4.1) $x_k \in (X \cap \mathcal{Z}) \setminus \mathcal{F}$ exist. Then, let us consider a limit point $\bar{x} \in (X \cap \mathcal{Z}) \setminus \mathcal{F}$ of the sequence $\{x_k\}$ and, without loss of generality, let us call the corresponding subsequence as $\{x_k\}$ as well. Then, since $x_k \to \bar{x}$, the discrete variables remain fixed, i.e. $(x_k)_i = \bar{x}_i$, for all $i \in I^z$ and k sufficiently large. Now, the proof continues by separately assuming that point (i) or (ii) of Assumption 7 holds.

First we assume that point (i) of Assumption 7 holds at $\bar{x}$. Therefore, a direction $\bar{s} \in D^c(\bar{x})$ exists such that

$$(\xi^{g_i})^\top \bar{s} < 0 \ \text{ for all } \ \xi^{g_i} \in \partial_c g_i(\bar{x}) \text{ with } i \in \{h \in \{1,2,\dots,m\} \,:\, g_h(\bar{x}) \ge 0\}.$$

In particular, it follows that

$$\max_{\substack{\xi^{g_i} \in \partial_c g_i(\bar{x}) \\ i \in I(\bar{x})}} (\xi^{g_i})^\top \bar{s} = -\eta < 0, \tag{A.1}$$

where $I(\bar{x}) = \{i \in \{1,2,\dots,m\} \,:\, g_i(\bar{x}) = \bar{\phi}(\bar{x})\}$, $\bar{\phi}(x) = \max\{0, g_1(x), g_2(x), \dots, g_m(x)\}$ and η is a positive scalar. Note that $\bar{x} \notin \mathcal{F}$ implies $\bar{\phi}(\bar{x}) > 0$.

By Proposition 9, it follows that $\bar{s} \in D^c(x_k)$. Moreover, since x_k satisfies the Definition 10 of stationary point, we have that

$$0 \leq P^{Cl}(x_k;\varepsilon,\bar{s}) = \max_{\xi \in \partial_c P(x_k;\varepsilon)} \xi^\top \bar{s}. \tag{A.2}$$

By [14], we know that

$$\partial_c P(x_k;\varepsilon) \subseteq \partial_c f(x_k) + \frac{1}{\varepsilon}\partial_c(\max\{0, g_1(x_k), g_2(x_k), \dots, g_m(x_k)\}) \tag{A.3}$$

and

$$\partial(\max\{0, g_1(x_k), g_2(x_k), \dots, g_m(x_k)\}) \subseteq Co\big(\{\partial_c g_i(x_k) \,:\, i \in I(x_k)\}\big), \tag{A.4}$$

where $\beta^i \geq 0$ for all $i \in I(x_k)$ and $Co(A)$ denotes the convex hull of a set A (see [47, Theorem 3.3]).

Therefore, by (A.2)–(A.4), $\xi_k^f \in \partial_c f(x_k)$, $\xi_k^{g_i} \in \partial_c g_i(x_k)$ and β_k^i with $i \in I(x_k)$ exist such that

$$\begin{aligned} &\left(\xi_k^f + \frac{1}{\varepsilon_k}\sum_{i \in I(x_k)} \beta_k^i \xi_k^{g_i}\right)^\top \bar{s} \geq 0, \\ &\sum_{i \in I(x_k)} \beta_k^i = 1 \text{ and } \beta_k^i \geq 0. \end{aligned} \tag{A.5}$$

Since m is a finite number, there exists a subsequence of $\{x_k\}$ such that $I(x_k) = \bar{I}$.

Then, recalling that $(x_k)_i = \bar{x}_i$, for all $i \in I^z$ and for k sufficiently large, and since a locally Lipschitz continuous function has a generalized gradient which is locally bounded, it results that the sequences $\{\xi_k^f\}$ and $\{\xi_k^{g_i}\}$, with $i \in \bar{I}$, are bounded. Hence, we get that

$$\xi_k^f \to \bar{\xi}^f, \tag{A.6a}$$

$$\xi_k^{g_i} \to \bar{\xi}^{g_i} \text{ for all } i \in \bar{I}, \tag{A.6b}$$

$$\beta_k^i \to \bar{\beta}^i \text{ for all } i \in \bar{I}. \tag{A.6c}$$

Now the upper semicontinuity of $\partial_c f$ and $\partial_c g_i$, with $i \in \bar{I}$, at $\bar{x}$ (see Proposition 2.1.5 in [14]) implies that $\bar{\xi}^f \in \partial_c f(\bar{x})$ and $\bar{\xi}^{g_i} \in \partial_c g_i(\bar{x})$ for all $i \in \bar{I}$.

The continuity of the problem functions guarantees that for k sufficiently large

$$\{i \,:\, g_i(\bar{x}) - \phi(\bar{x}) < 0\} \subseteq \{i \,:\, g_i(x_k) - \phi(x_k) < 0\},$$

and, in turn, this implies that for k sufficiently large

$$\{i \,:\, g_i(x_k) - \phi(x_k) = 0\} = I(x_k) \subseteq I(\bar{x}) = \{i \,:\, g_i(\bar{x}) - \phi(\bar{x}) = 0\}.$$

Since $I(x_k) \subseteq I(\bar{x})$, we have that

$$\bar{I} \subseteq I(\bar{x}). \tag{A.7}$$

Finally, for k sufficiently large, (A.1), (A.6), and (A.7) imply

$$\left(\xi_k^{g_i}\right)^\top \bar{s} \leq -\frac{\eta}{2} \quad \text{for all } i \in \bar{I}, \tag{A.8}$$

and (A.5), multiplied by ε_k, implies

$$\left(\varepsilon_k \xi_k^f + \sum_{i \in \bar{I}} \beta_k^i \xi_k^{g_i}\right)^\top \bar{s} \geq 0. \tag{A.9}$$

Equations (A.8) and (A.9) yield

$$0 \leq \left(\varepsilon_k \xi_k^f + \sum_{i \in \bar{I}} \beta_k^i \xi_k^{g_i}\right)^\top \bar{s} \leq \left(\varepsilon_k \xi_k^f\right)^\top \bar{s} - \frac{\eta}{2}.$$

which, by using (A.6), gives rise to a contradiction when $\varepsilon_k \to 0$.

Now we assume that point (*ii*) of Assumption 7 holds at $\bar{x}$. Let $\bar{d} \in D^z(\bar{x})$ be the direction such that

$$\sum_{i=1}^m \max\{0, g_i(\bar{x} + \bar{d})\} < \sum_{i=1}^m \max\{0, g_i(\bar{x})\}. \tag{A.10}$$

recalling that $(x_k)_i = \bar{x}_i$, for all $i \in I^z$ and for k sufficiently large, we have that for k sufficiently large $D^z(\bar{x}) = D^z(x_k)$, so that $\bar{d} \in D^z(x_k)$. By definition of stationary point and discrete neighborhood, we have

$$P(x_k;\varepsilon_k) \leq P(x_k + \bar{d};\varepsilon_k), \qquad \text{where } x_k + \bar{d} \in \mathcal{B}^z(x_k).$$

Hence,

$$f(x_k) + \frac{1}{\varepsilon_k}\sum_{i=1}^m \max\{0, g_i(x_k)\} \leq f(x_k + \bar{d}) + \frac{1}{\varepsilon_k}\sum_{i=1}^m \max\{0, g_i(x_k + \bar{d})\}.$$

Multiplying by ε and considering that $\varepsilon_k \to 0$, if we take the limit for $k \to \infty$ we have that

$$\sum_{i=1}^m \max\{0, g_i(\bar{x})\} \leq \sum_{i=1}^m \max\{0, g_i(\bar{x} + \bar{d})\}.$$

The latter equation is in contradiction with (A.10). □

Now, we can prove that there exists a threshold value $\bar{\varepsilon}$ for the penalty parameter such that, for any $\varepsilon \in (0, \bar{\varepsilon})$, any local minimum of the penalized problem is also a local minimum of the original problem. In particular, the following two propositions are analogous to [19, Theorems 10 and 11].

Proposition 11 *Let Assumptions 7 hold. Given Problem (1.3) and considering Problem (4.1), a threshold value $\bar{\varepsilon} > 0$ exists such that for every $\varepsilon \in (0, \bar{\varepsilon}]$, any local minimum point $\bar{x}$ of Problem (4.1) is also a local minimum of Problem (1.3).*

Proof Let $\bar{x}$ be any local minimum point of $P(x;\epsilon)$ on $X \cap \mathcal{Z}$. By Lemma 2, we have that $\bar{x}$ is also a Clarke stationary point.

Now Proposition 10 implies that a threshold value $\varepsilon^* > 0$ exists such that $\bar{x} \in \mathcal{F} \cap \mathcal{Z} \cap X$ for any $\varepsilon \in (0, \varepsilon^*]$. Therefore, $P(\bar{x};\varepsilon) = f(\bar{x})$ which implies that $\bar{x}$ is also a local minimum point for Problem 1.3. □

Proposition 12 *Let Assumption 7 hold. Given Problem (1.3) and considering Problem (4.1), a threshold value $\bar{\varepsilon} > 0$ exists such that for every $\epsilon \in (0, \bar{\varepsilon})$, any global minimum point $\bar{x}$ of Problem (4.1) is also a global minimum point of Problem (1.3) and viceversa.*

Proof We start by proving that any global minimum point of Problem (4.1) is also a global minimum point of Problem (1.3). Proceeding by contradiction, let us assume that for any integer k a positive scalar $\varepsilon_k < 1/k$ and a point x_k exist such that x_k is a global minimum point of $P(x_k;\varepsilon_k)$ but it is not a global minimum point of $f(x)$. If we denote as $\hat{x}$ a global minimum point of $f(x)$, we have that

$$P(x_k;\varepsilon_k) \leq P(\hat{x};\varepsilon_k) = f(\hat{x}). \tag{A.11}$$

Since x_k are global minimum points, by Lemma 2 they are also stationary points of $P(x;\varepsilon_k)$ according to Clarke definition for the continuous variables. By Proposition 10 there exists a threshold value $\varepsilon^* > 0$ such that $x_k \in \mathcal{F} \cap X \cap \mathcal{Z}$ for any $\varepsilon_k \in (0, \varepsilon^*]$. Therefore, $P(x_k;\varepsilon_k) = f(x_k)$. By (A.11), it follows that $f(x_k) \leq f(\hat{x})$, contradicting the assumption that x_k is not a global minimum point of $f(x)$.

Now we prove that any global minimum point $\bar{x}$ of Problem (1.3) is also a global minimum point of Problem (4.1) for any $\varepsilon \in (0, \bar{\varepsilon})$. Since $\bar{x} \in \mathcal{F} \cap \mathcal{Z} \cap X$, we have that $P(\bar{x};\varepsilon) = f(\bar{x})$. By the previous proof, a global minimizer x_ε of $P(x;\varepsilon)$ is feasible for Problem (1.3), hence $P(x_\varepsilon;\varepsilon) = f(x_\varepsilon)$. Furthermore, it is also a global minimum point of Problem (1.3), thus we have $f(x_\varepsilon) = f(\bar{x})$. Therefore, since $P(x_\varepsilon;\varepsilon) = f(\bar{x})$, $\bar{x}$ is also a global minimum point of $P(x;\varepsilon)$. □

In order to give stationarity results for Problem (4.1), we have the following proposition.

Proposition 13 *For any $\varepsilon > 0$, every stationary point $\bar{x}$ of Problem (4.1) according to Clarke, such that $\bar{x} \in \mathcal{F} \cap \mathcal{Z} \cap X$, is also a stationary point of Problem (1.3).*

Proof Since $\bar{x}$ is, by assumption, a stationary point of Problem (4.1) according to Clarke (see Lemma 2), then we have by definition of Clarke stationarity that for all $s \in D^c(\bar{x})$,

$$P^{Cl}(\bar{x};\varepsilon,s)=\max\left\{\xi^\top s\ :\ \xi\in\partial_c P(\bar{x};\varepsilon)\right\}\geq 0, \tag{A.12}$$

and

$$P(\bar{x};\varepsilon)\leq P(x;\varepsilon)\quad \text{for all } x\in\mathcal{B}^z(\bar{x}). \tag{A.13}$$

Hence, by A.12, there exists $\xi_s\in\partial_c P(\bar{x};\varepsilon)$ such that $(\xi_s)^\top s\geq 0$ for all $s\in D^c(\bar{x})$. Now, we recall that

$$\partial_c P(x;\varepsilon)\subseteq\partial_c f(x)+\frac{1}{\varepsilon}\sum_{i\in I(x)}\beta_i\partial_c g_i(x),$$

for some β_i, with $i\in I(x)$, such that $\sum_{i\in I(x)}\beta_i=1$ and $\beta_i\geq 0$ for all $i\in I(x)$. Hence, we have that $\xi_s\in\partial_c f(\bar{x})+\frac{1}{\varepsilon}\sum_{i\in I(\bar{x})}\beta_i\partial_c g_i(\bar{x})$. Then, denoting $\lambda_i=\beta_i/\varepsilon$ with $i\in I(\bar{x})$, and assuming $\lambda_i=0$ for all $i\notin I(\bar{x})$, we can write, for all $s\in D^c(\bar{x})$,

$$\max\left\{\xi^\top s\ :\ \xi\in\partial_c f(\bar{x})+\sum_{i=1}^m\lambda_i\partial_c g_i(\bar{x})\right\}\geq 0, \tag{A.14}$$

$$(\lambda)^T g(\bar{x})=0 \text{ and } \lambda\geq 0. \tag{A.15}$$

Recalling that $\bar{x}$ is feasible for Problem (1.3), by (A.13) we have

$$f(\bar{x})\leq f(x)\quad \text{for all } x\in\mathcal{B}^z(\bar{x}), \tag{A.16}$$

Considering that $\bar{x}\in\mathcal{F}\cap\mathcal{Z}\cap X$, (A.14), (A.15), and (A.16) prove that $\bar{x}$ is a KKT stationary point for Problem (1.3), thus concluding the proof. □

Proposition 14 *Let Assumption 7 hold. Then, a threshold value $\varepsilon^*>0$ exists such that, for every $\varepsilon\in(0,\varepsilon^*]$, every stationary point $\bar{x}$ of Problem (4.1) is stationary (according to Definition 13) for Problem (1.3).*

Proof Since $\bar{x}$ is stationary for Problem (4.1), we have by Definition 10 that

$$P^\circ(\bar{x};\varepsilon,s)\geq 0\ \text{ for all } s\in D^c(\bar{x}), \tag{A.17}$$

and

$$P(\bar{x};\varepsilon)\leq P(x;\varepsilon)\quad \text{for all } x\in\mathcal{B}^z(\bar{x}). \tag{A.18}$$

Then, by Definitions 3 and 9 , we have that

$$\begin{aligned}\limsup_{y_c\to\bar{x}_c,y_z=\bar{x}_z,t\downarrow 0}\frac{P(y+ts;\varepsilon)-P(y;\varepsilon)}{t}&=P^{Cl}(\bar{x};\varepsilon,s)\\ &\geq P^\circ(\bar{x};\varepsilon,s)\ \text{ for all } s\in D^c(\bar{x}).\end{aligned}$$

By (A.17), it follows that

$$P^{Cl}(\bar{x};\varepsilon,s) \geq 0 \quad \text{for all} \quad s \in D^c(\bar{x}).$$

The proof follows by considering Propositions 10, 13 and (A.18). □

Data availability The datasets generated during and/or analysed during the current study are available in the DFL repository, http://www.iasi.cnr.it/~liuzzi/dfl as package DFNDFL.

References

1. Abramson, M., Audet, C.: Filter pattern search algorithms for mixed variable constrained optimization problems. Pac. J. Optim. **3**, 1–10 (2004)
2. Abramson, M., Audet, C., Chrissis, J., Walston, J.: Mesh adaptive direct search algorithms for mixed variable optimization. Optim. Lett. **3**, 35–47 (2009). https://doi.org/10.1007/s11590-008-0089-2
3. Abramson, M., Audet, C., Dennis, J.E., Jr., Le Digabel, S.: OrthoMADS: a deterministic MADS instance with orthogonal directions. SIAM J. Optim. **20**, 948–966 (2009). https://doi.org/10.1137/080716980
4. Audet, C., Dennis, J.E., Jr.: Pattern search algorithms for mixed variable programming. SIAM J. Optim. (2001). https://doi.org/10.1137/S1052623499352024
5. Audet, C., Dennis, J.E., Jr.: A pattern search filter method for nonlinear programming without derivatives. SIAM J. Optim. (2004). https://doi.org/10.1137/S105262340138983X
6. Audet, C., Dennis, J.E., Jr.: Mesh adaptive direct search algorithms for constrained optimization. SIAM J. Optim. **17**, 188–217 (2006). https://doi.org/10.1137/040603371
7. Audet, C., Dennis, J.E., Jr.: A progressive barrier for derivative-free nonlinear programming. SIAM J. Optim. **20**(1), 445–472 (2009)
8. Audet, C., Hare, W.: Derivative-Free and Blackbox Optimization. Springer, New York (2017). https://doi.org/10.1007/978-3-319-68913-5
9. Audet, C., Le Digabel, S., Tribes, C., Montplaisir, V.R.: The NOMAD project. https://www.gerad.ca/nomad/
10. Audet, C., Dennis, J.E., Jr., Le Digabel, S.: Globalization strategies for mesh adaptive direct search. Comput. Optim. Appl. **46**, 193–215 (2010). https://doi.org/10.1007/s10589-009-9266-1
11. Audet, C., Le Digabel, S., Tribes, C.: The mesh adaptive direct search algorithm for granular and discrete variables. SIAM J. Optim. **29**, 1164–1189 (2019). https://doi.org/10.1137/18M1175872
12. Boukouvala, F., Misener, R., Floudas, C.: Global optimization advances in mixed-integer nonlinear programming, minlp, and constrained derivative-free optimization, cdfo. Eur. J. Oper. Res. (2015). https://doi.org/10.1016/j.ejor.2015.12.018
13. Bratley, P., Fox, B.L.: Algorithm 659: implementing Sobol's quasirandom sequence generator. ACM Trans. Math. Softw. **14**(1), 88–100 (1988). https://doi.org/10.1145/42288.214372
14. Clarke, F.: Optimization and Nonsmooth Analysis. Wiley, New York (1983)
15. Conn, A., Scheinberg, K., Vicente, L.N.: Introduction to Derivative-Free Optimization. MPS-SIAM Book Series on Optimization, SIAM, Philadelphia (2009)

16. Costa, A., Nannicini, G.: RBFOpt: an open-source library for black-box optimization with costly function evaluations. Math. Program. Comput. **10**, 597–629 (2018). https://doi.org/10.1007/s12532-018-0144-7
17. Custódio, A.L., Vicente, L.N.: Using sampling and simplex derivatives in pattern search methods. SIAM J. Optim. **18**, 537–555 (2007). https://doi.org/10.1137/050646706
18. Custódio, A.L., Dennis, J.E., Jr., Vicente, L.N.: Using simplex gradients of nonsmooth functions in direct search methods. IMA J. Numer. Anal. **28**(4), 770–784 (2008). https://doi.org/10.1093/imanum/drn045
19. Di Pillo, G., Facchinei, F.: Exact barrier function methods for Lipschitz programs. Appl. Math. Optim. **32**(1), 1–31 (1995)
20. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**(2), 201–213 (2002)
21. Fasano, G., Liuzzi, G., Lucidi, S., Rinaldi, F.: A linesearch-based derivative-free approach for nonsmooth constrained optimization. SIAM J. Optim. **24**, 959–992 (2014). https://doi.org/10.1137/130940037
22. Garcìa-Palomares, U.M., Rodriguez, J.F.: New sequential and parallel derivative-free algorithms for unconstrained minimization. SIAM J. Optim. **13**, 79–96 (2002). https://doi.org/10.1137/S1052623400370606
23. Garcìa-Palomares, U.M., Costa-Montenegro, E., Asorey Cacheda, R., González-Castaño, F.: Adapting derivative free optimization methods to engineering models with discrete variables. Optim. Eng. (2012). https://doi.org/10.1007/s11081-011-9168-9
24. Gould, F.J., Tolle, J.W.: Geometry of optimality conditions and constraint qualifications. Math. Program. (1972). https://doi.org/10.1007/BF01584534
25. Halstrup, M.: Black-box optimization of mixed discrete-continuous optimization problems (2016). Retrieved from Eldorado - Repository of the TU Dortmund
26. Halton, J.H.: On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. Num. Math. **2**(1), 84–90 (1960)
27. Hemker, T., Fowler, K., Farthing, M., Von Stryk, O.: A mixed-integer simulation-based optimization approach with surrogate functions in water resources management. Optim. Eng. **9**, 341–360 (2008). https://doi.org/10.1007/s11081-008-9048-0
28. Jahn, J.: Introduction to the Theory of Nonlinear Optimization, 3rd edn. Springer, Incorporated, New York (2014)
29. Karmitsa, N.: Test problems for large-scale nonsmooth minimization. Reports of the Department of Mathematical Information Technology, Series B, Scientific computing, B 4/2007 (2007)
30. Larson, J., Menickelly, M., Wild, S.M.: Derivative-free optimization methods. Acta Num. **28**, 287–404 (2019). https://doi.org/10.1017/s0962492919000060
31. Larson, J., Leyffer, S., Palkar, P., Wild, S.M.: A method for convex black-box integer global optimization. J. Glob. Optim. **1**, 1–39 (2021). https://doi.org/10.1007/s10898-020-00978-w
32. Le Digabel, S.: Algorithm 909: NOMAD: Nonlinear optimization with the MADS algorithm. ACM Trans. Math. Softw. **37**(4), 1–15 (2011)
33. Liuzzi, G., Lucidi, S., Rinaldi, F.: Derivative-free methods for bound constrained mixed-integer optimization. Comput. Optim. Appl. (2012). https://doi.org/10.1007/s10589-011-9405-3
34. Liuzzi, G., Lucidi, S., Rinaldi, F.: Derivative-free methods for mixed-integer constrained optimization problems. J. Optim. Theory Appl. (2014). https://doi.org/10.1007/s10957-014-0617-4
35. Liuzzi, G., Lucidi, S., Rinaldi, F.: An algorithmic framework based on primitive directions and nonmonotone line searches for black-box optimization problems with integer variables. Math. Program. Comput. **12**, 673–702 (2020). https://doi.org/10.1007/s12532-020-00182-7
36. Lucidi, S., Sciandrone, M.: On the global convergence of derivative-free methods for unconstrained optimization. SIAM J. Optim. **13**, 97–116 (2002). https://doi.org/10.1137/S1052623497330392
37. Lucidi, S., Piccialli, V., Sciandrone, M.: An algorithm model for mixed variable programming. SIAM J. Optim. (2005). https://doi.org/10.1137/S1052623403429573
38. Luǩsan, V., Vlček, J.: Test problems for nonsmooth unconstrained and linearly constrained optimization. Technical report VT798-00, Institute of Computer Science, Academy of Sciences of the Czech Republic (2000)
39. Mangasarian, O., Fromovitz, S.: The fritz john necessary optimality conditions in the presence of equality and inequality constraints. J. Math. Anal. Appl. **17**(1), 37–47 (1967)

40. Moré, J., Wild, S.: Benchmarking derivative-free optimization algorithms. SIAM J. Optim. **20**, 172–191 (2009). https://doi.org/10.1137/080724083
41. Müller, J.: MISO: mixed-integer surrogate optimization framework. Optim. Eng. **17**, 1–27 (2015). https://doi.org/10.1007/s11081-015-9281-2
42. Müller, J., Shoemaker, C., Piché, R.: SO-I: A surrogate model algorithm for expensive nonlinear integer programming problems including global optimization applications. J. Glob. Optim. (2013). https://doi.org/10.1007/s10898-013-0101-y
43. Müller, J., Shoemaker, C.A., Piché, R.: SO-MI: A surrogate model algorithm for computationally expensive nonlinear mixed-integer black-box global optimization problems. Comput. Oper. Res. **40**(5), 1383–1400 (2013). https://doi.org/10.1016/j.cor.2012.08.022
44. Newby, E., Ali, M.: A trust-region-based derivative free algorithm for mixed integer programming. Comput. Optim. Appl. **60**, 199–229 (2014). https://doi.org/10.1007/s10589-014-9660-1
45. Porcelli, M., Toint, P.: BFO, a trainable derivative-free brute force optimizer for nonlinear bound-constrained optimization and equilibrium computations with continuous and discrete variables. ACM Trans. Math. Softw. **44**, 1–25 (2017). https://doi.org/10.1145/3085592
46. Powell, M.: The BOBYQA algorithm for bound constrained optimization without derivatives. Technical Report, Department of Applied Mathematics and Theoretical Physics (2009)
47. Rockafellar, R.T.: Convex Analysis. Princeton University Press, Princeton (1970)
48. Sobol, I.: Uniformly distributed sequences with an additional uniform property. USSR Comput. Math. Math. Phys. **16**(5), 236–242 (1976). https://doi.org/10.1016/0041-5553(76)90154-3
49. Sriver, T.A., Chrissis, J.W., Abramson, M.A.: Pattern search ranking and selection algorithms for mixed variable simulation-based optimization. Eur. J. Oper. Res. **198**(3), 878–890 (2009). https://doi.org/10.1016/j.ejor.2008.10.020
50. Torczon, V.: On the convergence of pattern search algorithms. SIAM J. Optim. **7**(1), 1–25 (1997). https://doi.org/10.1137/S1052623493250780
51. Vicente, L.N., Custódio, A.L.: Analysis of direct searches for discontinuous functions. Math. Program. **133**, 1–27 (2009). https://doi.org/10.1007/s10107-010-0429-8
52. Yang, S., Liu, H., Pan, C.: An efficient derivative-free algorithm for bound constrained mixed-integer optimization. Evol. Intell. (2019). https://doi.org/10.1007/s12065-019-00326-2