Computational Optimization and Applications (2023) 85:821–856
https://doi.org/10.1007/s10589-023-00469-0

Convergence of derivative-free nonmonotone Direct Search Methods for unconstrained and box-constrained mixed-integer optimization

Ubaldo M. García Palomares[1]

Received: 30 September 2019 / Accepted: 22 February 2023 / Published online: 25 April 2023

Abstract
This paper presents a class of nonmonotone Direct Search Methods that converge to stationary points of unconstrained and boxed constrained mixed-integer optimization problems. A new concept is introduced: the quasi-descent direction. A point x is stationary on a set of search directions if there exists no feasible qdd on that set. The method does not require the computation of derivatives nor the explicit manipulation of asymptotically dense matrices. Preliminary numerical experiments carried out on small to medium problems are encouraging.

Keywords Non monotone · Direct Search Methods · Box constraints · Mixed-integer variables · Quasi descent direction · Derivative free

1 Introduction

Direct Search Methods (DSM) have been extensively used for solving optimization problems with continuous variables. Despite their simplicity, encouraging numerical results have been reported for solving small and medium size problems, with or without first order derivative information (see [22] and references therein). DSMs solve the minimization problem, minimize $f(x), x \in \mathcal{F} \subseteq \mathbb{R}^n$ as follows: At the *i-th* iteration an estimate $x_i \in \mathcal{F}$ to the solution is known. A search direction $d_i \in \overset{\infty}{D}= \{d \in \mathbb{R}^n : ||d|| = 1\}$ and a stepsize $\tau_i \in \mathbb{R}_+$ complying with manageable conditions are found, and a new estimate $x_{i+1} = x_i + \tau_i d_i \in \mathcal{F}$ is generated. Early application of DSMs on optimization problems with continuous variables were monotone; i.e. they forced a sufficient decrease to the function values, namely $f(x_{i+1}) \leq f(x_i) - \sigma_i$; for some $\sigma_i > 0$. Steepest descent, variable metric and Newton methods are classical DSM instances which use or approximate first order information.

✉ Ubaldo M. García Palomares
ubaldo@gti.uvigo.es

1 Universidade de Vigo, Vigo, Spain

The pattern search method (PSM), introduced in [42], is an instance of a monotone DSM with no derivative information. An extensive list of references for monotone algorithms is given in [5]. Reference [23] stands as the pioneer work of nonmonotone DSMs to problems with continuous variables. Several variants were suggested which inherit the properties of Newton's method with explicit derivative information [11, 14, 24, 44]. These authors claim that nonmonotone DSMs might improve the performance of a monotone algorithm, when the estimates enter into a curved narrow valley. In smooth constrained optimization, nonmonotone algorithms are used to avoid the Maratos' effect [45, and references therein]. Performance of nonmonotone DSMs is more stable on noisy functions than its monotone counterpart [5, 16]. Authors in references [14, 16, 17, 19, 20] employ nonmonotone DSMs to *try to* avoid convergence to nearby local minimizers. Finally, the authors in [21] claim that almost any *deterministic* monotone optimization algorithm for solving models with continuous variables has a nonmonotone counterpart. This paper analyzes the basic theory needed for solving $\text{minimize}_{(x;z)\in\mathcal{F}}\, g(x;z)$ with a non-monotone DSM, where $\mathcal{F}$ is an unspecified feasible set. The results are then adapted to solve the mixed integer optimization problem (1) formulated as

$$\underset{(x;z)\in\mathcal{F}}{\text{minimize}}\ g(x;z) : \mathbb{R}^{n+p} \to \mathbb{R}, \tag{1a}$$

$$\mathcal{F} = \{x \in \mathbb{R}^n,\ z \in \mathbb{Z}^p : s \le (x;z) \le t\}, \tag{1b}$$

where $g(\cdot)$ might be nonsmooth.

To facilitate our exposition, $(x;z)$ stands for a column vector of $n+p$ components, with $x \in \mathbb{R}^n$ and $z \in \mathbb{Z}^p$. Thus, in (1), s and t are known vectors in $\mathbb{R}^{n+p}$, which represent respectively, the lower and upper bounds of all variables in (1). An initial effort to use PSMs for solving nonlinearly constrained problems with discrete *-and/or categorical-* variables can be found in [1, 5]. This paper is an outgrowth of the nonmonotone DSM proposed in [19], where the discrete variables were assumed to lay on a regular grid of discrete points.

This work presents a convergence theory for a class of monotone and nonmonotone DSMs, which allows us to have at hand a *common* framework for solving problem (1) and its particular instances: unconstrained and box-constrained optimization problems with mixed integer variables. We assume that we use penalization, Lagrangian functions, filtering, or any other technique that may transform a constrained problem into a single or a sequence of box-constrained problems structured like (1). The authors in [1, 2] propose a simple barrier function, which assumes that the objective function is infinite at all infeasible points. Since there exist nonsmooth functions that do not decrease along any direction from a point that is not a minimum [13, Exercise 2.6], we always consider a finite set of search directions; nonetheless, we show that our algorithm might detect directions of negative curvature. We solve the mixed optimization problem (1) under relaxed conditions, regardless of the availability of derivative information. Some degree of randomness is necessary to fully exploit the algorithm's capabilities, though the numerical tests show that a deterministic version of the algorithm performs well.

For nonsmooth problems, the nonmonotone DSM proposed in this paper shares convergence properties with monotone DSMs frequently cited. As our purpose is to apply this methodology in derivative free optimization (DFO), where differentiability and other function features may be unknown to the user, no effort is devoted to the rate of convergence.

The paper's primary goal is to analyze non-monotone DSM algorithms, which in general, have been scarcely considered in the open literature. All convergence results hold with either explicit use of first order derivatives or for DFO. Moreover, if first order information is at hand, the algorithms simplify significantly, and, generally, a singleton search direction is easily identified.

The remainder of this paper is organized as follows. In the next section we prove some lemmas that are essential for the understanding of the algorithms. We state some useful definitions and display pseudocodes that will guide us in the analysis of the optimization problems to be exposed in Sects. 3, 4 and 5. Section 3 describes the DSM proposed for solving problems with continuous variables. We show that non-monotone DSMs generate a sequence onverging to a point satisfying a *non-smooth stationarity condition*; which, to our knowledge, has been overlooked. For Lipschitzian functions the sequence converges to a point satisfying the stationarity condition in the Clarke sense. Section 3 conforms the theory developed in Sect. 2 to unconstrained and box-constrained problems. Compactness, Fréchet differentiability and continuous first order derivatives around limit points are sufficient conditions to achieve convergence. In Sects. 4 and 5 we assume that variables can take discrete values. The methodology persists. Convergence to a stationarity point does not require to detect the best local value in the vicinity of a solution estimate. We suggest a variable separation strategy to solve Problem (1). In Sect. 6 we solve a group of low-dimensional academic problems. The results look promising, and seem to validate the theoretical findings. We also solve a real application problem and report a new global solution to the problem. The paper concludes with some remarks and open questions to research topics that are under investigation.

In short, and for the sake of clarity, we analyze convergence of the algorithms that solve the basic unconstrained and box-constrained optimization problems. This analysis is in substance carried over to the mixed variable problem (1), which is formally studied in Sect. 5.

Notation Throughout the paper we use a rather standard notation with some peculiarities.

- $\mathbb{R}^n$, $\mathbb{R}^p$, $\mathbb{R}^r$, and so forth, are Euclidean spaces.
- $\mathcal{F} \subseteq \mathbb{R}^n$ is the feasible set,
- superscripts stand for components and sub-indexes represent different vectors,
- $d^T w$ is the inner product of vectors d, w defined in the same Euclidean space,
- uu^T is a matrix U with components $U^{jk} = u^j u^k$;
- $(x; z)$ is a vector with $(n + p)$ components, $x \in \mathbb{R}^n$, $z \in \mathbb{Z}^p$,
- When $\hat{z}$ is fixed, $f(x)$ denotes $g(x; \hat{z})$.
  Likewise, When $\hat{x}$ is fixed, $f(z)$ denotes $g(\hat{x}; z)$.
- $\nabla f(x)$ is the gradient,

- $f^0(\hat{x}, \hat{d}) \overset{\Delta}{=} \limsup\limits_{\substack{x \to \hat{x} \\ \tau \to 0^+}} \dfrac{f(x + \tau \hat{d}) - f(x)}{\tau}$ is the Clarke generalized directional derivative at $\hat{x}$ along $\hat{d}$,
- $g(x; z) : \mathbb{R}^{n+p} \to \mathbb{R}, x \in \mathbb{R}^n, z \in \mathbb{Z}^p$.
- $g_x^0(\hat{x}; \hat{z}, \hat{d}) \overset{\Delta}{=} \limsup\limits_{\substack{x \to \hat{x} \\ \tau \to 0^+}} \dfrac{g(x + \tau \hat{d}; \hat{z}) - g(x; \hat{z})}{\tau}$,
- $\sigma(\cdot) : \mathbb{R}_+ \to \mathbb{R}_+$ stands for a *sigma-function*, which is:

$$\begin{aligned}&\text{forcing: } (\sigma(\tau_i) \to 0) \text{ iff } (\tau_i \to 0),\\ &\text{unbounded from above: } (\sigma(\tau_i) \to \infty) \text{ iff } (\tau_i \to \infty), \text{ and}\\ &\text{little o}(\tau^2) : \lim_{\tau \to 0} \sigma(\tau)/\tau^2 = 0,\end{aligned}$$

- q represents a dummy integer, that plays different roles in the paper,
- I is the identity matrix and e_j, $j = 1, \ldots, n$ are its columns,
- H is the Househlder matrix $I - \dfrac{2}{u^T u} u u^T$, $u \neq 0$,
- $\overset{\infty}{D} = \{d \in \mathbb{R}^n : ||d|| = 1\}$,
- Other Capital letters are matrices or finite sets,
- If S is a finite set, $\#S$ is its cardinality,
- i will be the iteration number. It is usually implicitly assumed.
- We use the Matlab notation to invoke an algorithm. $[w_1, \ldots, w_q] = Algname(v_1, \ldots, v_p)$ invokes function ALGNAME() with p inputs and q outputs.

2 Preliminaries

This work addresses non-smooth functions. However, convergence theory might require some degree of differentiability. Some algorithms assume that $g(x; z)$ is Lipschitz continuous; which implies that the Clarke generalized directional derivative exists and it is finite. The Clarke generalized directional derivative is a useful concept in optimization theory.

In this section we define concepts and prove several lemmas needed in the general analysis of our algorithms. The basic convergence assumptions are:
A1: The objective function $g(x; z)$ is continuous and computable at all feasible points. The level set $L = \{(x; z) \in \mathcal{F} : g(x; z) \leq \varphi_0\}$ is compact for any φ_0.

This seems to be the weakest condition required by optimization algorithms that solve (1).
A2: For a fixed z, the objective function $g(x; z)$ in (1) is Fréchet differentiable and continuously differentiable around limit points.

A3: $g(x; z)$ is Lipschitz continuous in its first argument; that is, for a fixed z, $|g(y; z) - g(x; z)| \leq \vartheta ||y - x||$ for some $\vartheta > 0$.

The algorithm for solving the mixed integer problem (1) inherits most of its properties from the algorithms specifically proposed in Sects. 3 and 4 for solving the *pure* versions.

Definitions 1, 2 and 3 implicitly assume that $(\hat{x}; \hat{z}) \in \mathcal{F}$.

Definition 1 The direction $\hat{d}$ is a feasible direction at $(\hat{x}; \hat{z})$ when

$$(\hat{x} + \lambda\hat{d}; \hat{z}) \in \mathcal{F} \text{ for all sufficiently small } \lambda > 0. \tag{2}$$

Definition 2 The direction $\hat{d} \in \mathbb{R}^n$ is a quasi-descent direction (qdd) at $(\hat{x}; \hat{z})$ when

$$g(\hat{x} + \tau\hat{d}; \hat{z}) - g(\hat{x}; \hat{z}) \leq -\sigma(\tau) \text{ for all sufficiently small } \tau > 0. \tag{3}$$

Definition 3 The direction d is a feasible qdd at $(\hat{x}; \hat{z})$ when both (2) and (3) hold.

To facilitate the writing and the reading of this paper, we often denote $f(x) \overset{\Delta}{=} g(x; \hat{z})$ when $\hat{z}$ remains fixed.

Lemma 1 *Let* **A1, A2** *hold and let the unit vector d be a strictly descent direction at x, that is,* $d^T\nabla f(x) \leq -\alpha||\nabla f(x)||$, *for some* $\alpha > 0$. *Under these conditions d is a qdd.*

Proof It is obvious. $$\begin{aligned} f(x + \tau d) - f(x) &= \tau d^T\nabla f(x) + o(\tau) \\ &\leq -\tau\alpha||\nabla f(x)|| + o(\tau) \end{aligned}$$
The proof is complete if we define $\sigma(\tau) = \tau\alpha||\nabla f(x)|| - o(\tau)$. □

Lemma 2 *Under Assumptions* **A1–A3**, *let* $\{x_j\} \to \hat{x} \in \mathbb{R}^n$ *and* $f^0(\hat{x}, \hat{d}) < 0$, *then* $\hat{d}$ *is a qdd at some x sufficiently close to* $\hat{x}$.

Proof By **A3**, $f^0(\hat{x}, \hat{d})$ exists and it is finite. If $\hat{d}$ is not a qdd for all x close to $\hat{x}$, we can identify a sequence $\{\tau_j\}_{j\in\mathbb{N}} \downarrow 0$, $\{x_j\} \to \hat{x}$ for which

$$\frac{f(x_j + \tau_j\hat{d}) - f(x_j)}{\tau_j} > -\sigma(\tau_j)/\tau_j. \tag{4}$$

By taking limits, it follows that $f^0(\hat{x}, \hat{d}) \geq 0$, a contradiction. □

Corollary 1 *If* $\{x_i\} \to \hat{x}$ *and* $f^0(\hat{x}, d) \leq \alpha < 0$, *then d is a qdd for all x sufficiently close to* $\hat{x}$.

Lemma 3 *Let* $\hat{d} \in \overset{\infty}{D} = \{d \in \mathbb{R}^n : ||d|| = 1\}$ *be a feasible qdd at* $\hat{x} \in \mathcal{F}$. *Under Assumption* ***A1***, *there exist* $\tau > 0, \gamma > 0$ *such that*

$$\begin{gathered} \hat{x} + \tau\hat{d} + \gamma\tau\hat{d} \notin \mathcal{F} \\ \textbf{OR} \\ f(\hat{x} + \tau\hat{d} + \gamma\tau\hat{d}) - f(\hat{x} + \tau\hat{d}) > -\sigma(\tau). \end{gathered} \tag{5}$$

Proof Since $\hat{d}$ is a feasible qdd, (2–3) hold by definition for some $\hat{\tau} > 0$. If (5) holds for $\tau = \hat{\tau}$, and some $\hat{\gamma} > 0$, the lemma is valid. We now proceed by contradiction: If the statement of the lemma is false, Algorithm 1 generates an infinite loop with a sequence of feasible points with strictly decreasing $f(\cdot)$ values, which contradicts **A1**. □

```
Given $\hat{x} \in \mathcal{F}$, $\hat{\tau} > 0$, $\hat{d} \in D = \{d_1, \ldots, d_q\}$, satisfying (2-3),
this pseudo-code is an infinite loop when
$f(\cdot)$ is unbounded from below on a ray $d \in D$.
.....................................................................
$\hat{x} = \hat{x} + \hat{\tau}\hat{d}$
Choose any $\gamma > 0$, $j = 1$
WHILE $j \leq q$
    IF $(\hat{x} + \gamma\hat{\tau}d_j \in \mathcal{F})$ and $\big(f(\hat{x} + \gamma\hat{\tau}d_j) - f(\hat{x}) \leq -\sigma(\gamma\hat{\tau})\big)$
        $\hat{x} = \hat{x} + \gamma\hat{\tau}d_j$
    END- IF
    $j = j + 1$
END- WHILE
```

Algorithm 1 Function $f(\cdot)$ is unbounded from below.

We would like to state that a point $\overset{*}{x}$ is non-smooth stationary (nss) on D when there exists no feasible qdds at $\overset{*}{x}$; that is, when

$$\left.\begin{matrix}\lambda > 0\\ d \in D\end{matrix}\right\rangle \Rightarrow \big(\overset{*}{x} + \lambda d \notin \mathcal{F}\big) \ \mathbf{OR} \ \big(f(\overset{*}{x} + \lambda d) - f(\overset{*}{x}) > -o(\lambda)\big), \tag{6}$$

where $D \subseteq \overset{\infty}{D} = \{d \in \mathbb{R}^n : ||d|| = 1\}$.

Corollary 2 *Under Assumption* **A1**, *any nonempty feasible set* $\mathcal{F}$ *has an nss point satisfying* (6) *with* $D = \overset{\infty}{D}$.

Proof If there exists no nss, then given any $x \in \mathcal{F}$ there exists $\lambda > 0$, $d \in \overset{\infty}{D}$ and a qdd defined by (3). The WHILE loop in Algorithm 1 would generate an infinite sequence of strictly descent $f(\cdot)$, contradicting **A1**. □

The implication (6) is, in general, difficult to implement. We resort to finite sets D and discrete λ-values. We define a Λ-set as follows:

Definition 4 A Λ-set is a set of λ-values $\{\lambda_0, \lambda_1, \ldots, \}$ with the following properties:

- it is a bounded set,
- it contains an infinite number of distinct elements in $\mathbb{R}_+$ that can be lexicographically ordered; that is, $(j < k) \Rightarrow (\lambda_j > \lambda_k)$.
- its elements $\lambda_0, \lambda_1, \ldots$ converge to 0, that is, $\{\lambda_j\} \downarrow 0$.

A typical Λ-set that is often implicitly used by the optimization community is given by (7). Given $0 < \mu_s < \mu_t < 1$,

$$\text{Pick } \mu \in [\mu_s \ \mu_t], \lambda_0 > \varepsilon, \ \hat{\Lambda} = \{\lambda \in \mathbb{R}_+ : \lambda = (\mu)^k\lambda_0, k = 0, 1, 2, \ldots, \}. \tag{7}$$

Definition 5 The point $\overset{*}{x} \in \mathcal{F}$ is nss on (Λ, D) when

$$\left.\begin{array}{c}\lambda \in \Lambda, \lambda > \varepsilon \\ d \in D\end{array}\right\rangle \Rightarrow \left(\overset{*}{x} + \lambda d \notin \mathcal{F}\right) \textbf{ OR } \left(f(\overset{*}{x} + \lambda d) - f(\overset{*}{x}) > -o(\lambda)\right). \tag{8}$$

Definition 6 The point $\overset{*}{x} \in \mathcal{F}$ is totally stationary when (8) holds on $\overset{\infty}{D} = \{d \in \mathbb{R}^n : ||d|| = 1\}$.

For practical convenience, stationarity is defined on the sets (Λ, D). Ideally, we would like $\{x_i\}$ to converge to a total nss point. We will prove below that in unconstrained minimization this can occur on a very large set, provided that $\{D_i\}$ are carefully constructed for large enough i. On the other hand, a singleton direction is enough to ensure stationarity of smooth functions with derivative information.

Algorithm 1 is just one way to prove Lemma 3. Many other alternatives are possible, but this algorithm is easy to implement, and it is close to its monotone counterpart algorithm proposed in [18] and the related nonmonotone versions developed afterwards [14, 16, 17].

The following lemmas will be useful for differentiable functions.

Lemma 4 *Let $\{d_1, \cdots, d_n\}$ be a finite set of n orthogonal directions with $||d_j|| = 1, j = 1, \cdots, n$, and let $D \supseteq \{\pm d_1, \cdots, \pm d_n\}$. It follows that*

$$\forall w \in \mathbb{R}^n \left(\exists d \in D \colon d^T w \leq -(1/\sqrt{n})||w||\right). \tag{9}$$

Moreover, if $f(\cdot)$ is Fréchet differentiable,

$$\exists d \in D \colon d^T \nabla f(x) \leq -(1/\sqrt{n})||\nabla f(x)||. \tag{10}$$

Proof (9) is a well known fact and its proof is omitted. (10) is an instance of (9). □

Lemma 5 *Let $\hat{x} \in \mathbb{R}^n$ be a fixed point, let D be a set of search directions fulfilling the conditions required by Lemma 4 and let $\tau_0 \in \mathbb{R}_+$ be a positive number. If $f(\cdot)$ is differentiable and*

$$\left\langle\begin{array}{c} d \in D \\ \tau \in (0\ \tau_0)\end{array}\right\rangle \Rightarrow \left(f(\hat{x} + \tau d) - f(\hat{x}) > -o(\tau)\right) \tag{11}$$

then $\nabla f(\hat{x}) = 0$.

Proof We proceed by contradiction and assume that $\nabla f(\hat{x}) \neq 0$. Lemma 4 ensures that $d^T \nabla f(\hat{x}) \leq -(1/\sqrt{n})||\nabla f(\hat{x})||$ for some $d \in D$. It follows by Lemma 1 that d is a qdd; consequently, $\tau \in (0,\ \tau_0)$ exists satisfying (5); therefore (11) can occur only if $\nabla f(\hat{x}) = 0$. □

If $\nabla f(\hat{x}) = 0$, it is not possible to find a direction of descent satisfying $d^T \nabla f(\hat{x}) < 0$. Next lemma shows that, under more stringent conditions, it is possible to identify directions with negative curvature.

Lemma 6 *If* $\lim_{\lambda\to 0}\sigma(\lambda)/\lambda^2 = 0$, *and* $f(\cdot)\in C^2$, *the inequality* (3) *holds for directions with negative curvature.*

Proof If $\nexists\big(\lambda \in (0\ \lambda_0)\big)$ satisfying (3) and $\nabla f(x) = 0$, then for all $\lambda \in (0,\ \lambda_0)$, it follows that

$$\frac{1}{2}d^T\nabla^2 f(x)d + \frac{o(\lambda^2)}{\lambda^2} = \frac{f(x+\lambda d) - f(x)}{\lambda^2} \geq \frac{f(x+\lambda d) - \varphi}{\lambda^2} > -\frac{\sigma(\lambda)}{\lambda^2};$$

but this cannot hold for small enough λ when $d^T\nabla^2 f(x)d < 0$. □

2.1 The search directions

In the forthcoming sections we will see that the convergence analysis of smooth functions is simplified when (12) holds. So, we always force (12) to be valid when $\nabla f(\cdot)$ exists.

$$(\forall D_i)(\exists d \in D_i : d^T\nabla f(x_i) \leq -\alpha||\nabla f(x_i)||),\quad \text{for some } \alpha > 0. \tag{12}$$

By Lemma 4, orthogonal directions implicitly enforce (12) with $\alpha = 1/\sqrt{n}$. It is also known that the *cosine* value $\alpha = 1/\sqrt{n}$ cannot be improved by any D that spans $\mathbb{R}^n$ positively, with either $n+1$ or $2n$ search directions [35]. It has been argued that performance of a DFO algorithm may improve with $n+1$ search directions [4]. We suggest the directions $\pm d_1,\ldots,\pm d_n$, where $d_1,\ldots,d_n$ are defined by Algorithm 2, Line 4. They are the columns of the Householder matrix $H = I - 2uu^T$ generated by a unit random vector $u \in \mathbb{R}^n$, as suggested by [18]. A list of nice properties of $D = \{d_1,\ldots,d_n\}$ is:

```
[D] = ORTHOGONAL($s, x, t, \beta$)
Input: $s, x, t \in \mathbb{R}^n,\ x \in [s\ t], \beta > 0$.
Output: $n$ orthogonal directions in $\mathbb{R}^n$.
...........................................................
        FOR $j = 1, \ldots, n$
2a.         IF $\big((x^j \leq s^j + \beta)$ OR $(x^j \geq t^j - \beta)\big)$
2b.             $u^j = 0$
2c.         ELSE $\big(u^j \in [-1\ 1]\big)$      random
2d.         END- IFELSE
        END- FOR
3a.     $\zeta = u^T u$
3b.     IF ($\zeta < 0.001$)                  correction
3c.         $u = 0,\ \zeta = 1$
3d.     END- IF
4.      $d_j = e_j - \big(2u^j/\zeta\big)u, \quad j = 1, \ldots, n$
        ----------------------------------------
        RETURN $\{d_1, \ldots, d_n\}$
```

Algorithm 2 ORTHOGONAL() Constructs n unit orthogonal search directions $\{d_1,\ldots,d_n\}$ using the Householder matrix. $\beta = -\infty$ in unconstrained minimization. $\beta > 0$ in box constrained minimization.

- D is a set of orthogonal search directions.
- Algorithm 2 describes an *easy* way to generate random orthogonal directions obtained from the columns of the Householder matrix.
- It has been argued that some degree of randomness may benefit the performance of a *deterministic* algorithm [22]. In [6] the authors claim that they did not find a deterministic strategy to improve their algorithm.
- It is relatively simple to distribute the workload among processors [18].
- Line 4 of Algorithm 2 defines $d_j = He_j = e_j - (2u^j/u^T u)u,\ j = 1, \ldots, n$. It is not necessary to construct explicitly the whole matrix H. The vector u contains the information needed to generate the search directions. This feature allows a significant saving in memory for medium and large problems.
- Algorithm 2 conforms the search directions to the geometry of the constraints. It merely assigns $u^j = 0$ in Line 2b whenever the variable x^j is close to either one of its bounds. Note that

$$\left[u^j = 0\right] \Rightarrow \left[d_j = e_j,\ \text{and}\ d_k^j = 0, k \neq j\right]. \tag{13}$$

  Hence, $x^j + d_k^j = x^j$, for all $k \neq j$, which prevents the j-th variable from getting closer to its bounds.
- The published numerical results since its inception in [18] have been highly competitive [14, 16, 17, 19, 20].
- u^j can be randomly generated in $[-1\ 1]$, $j = 1, \ldots, n$. This is convenient when $x \in \mathbb{R}^n$.
- A suitable vector $u \in \mathbb{Z}^n$ can be as well randomly generated to handle integer variables. From Lines 3a and 4 of Algorithm 2 we obtain that $\zeta d_j = \zeta e_j - (2u^j)u$, where $\zeta = u^T u$ and $d_j = He_j$. It follows that

$$u \in \mathbb{Z}^n \Rightarrow \left(\tau d \in \mathbb{Z}^n \text{ for } d \in D \text{ and } \tau = \pm\zeta,\ \pm 2\zeta, \ldots\right). \tag{14}$$

2.2 The essential algorithm

The Iteration() algorithm, identified as Algorithm 3, will be the core iteration for those algorithms dealing with variables in $\mathbb{R}^n$. It is invoked by $[x_i,\ \varphi_i,\ \tau_i,\ D_i]$ = Iteration(x_{i-1}, φ_{i-1}). Iterative calls to Iteration() are carried out until convergence criteria are met. The formal input parameters of Iteration() are the actual estimate $x_{i-1} \in \mathbb{R}^n$ and $\varphi_{i-1} \in \mathbb{R}$, an upper bound of $f(x_{i-1})$. A new estimate x_i, a new functional upper bound φ_i, a stepsize τ_i, and the set of search directions D_i are returned by Iteration().

The Iteration()'s task is to find a feasible qdd. Lines 2a-2c check if the input estimate x_{i-1} is a feasible qdd; in which case, $\lambda \leq \varepsilon$ at Line 2d and Iteration() returns. Strictly speaking, x_{i-1} has no feasible qdd when for any $\lambda > 0$ and any $d \in \overset{\infty}{D}$, it follows that

$$x_{i-1} + \lambda d \notin \mathcal{F} \ \textbf{or}\ \ f(x_{i-1} + \lambda d) > \varphi_{i-1} - \sigma(\lambda). \tag{15}$$

```
$[x', \varphi', \tau', D'] = \text{ITERATION}(x, \varphi)$
Input: $x \in \mathcal{F}, \varphi \geq f(x)$
Output: $x' \in \mathcal{F}, \varphi' \geq f(x'), \tau' \in \mathbb{R}_+$, AND
$D' = \{d_1, \ldots, d_q\}$, $q$ directions in $\mathbb{R}^n$
.........................................................................
1a  Define $\Lambda$ (and $\mu$) satisfying (7)
1b  Choose $\lambda \in \Lambda$
1c  Construct $Q'$ by Algorithm 2 and $D' \supseteq Q'$
    WHILE ($\lambda > \varepsilon$)
2a      $y_j = x + \lambda d_j, \quad d_j \in D' = \{d_1, \ldots, d_j, \ldots\}$
2b      IF $(y_j \in \mathcal{F}$ and $f(y_j) \leq \varphi - \sigma(\lambda))$ BREAK
2c      $\lambda = \mu\lambda$
    END- WHILE
2d  IF ($\lambda \leq \varepsilon$)
2e      RETURN$[x, \varphi, \varepsilon, D']$
    END- IF
    Find $x' \in \mathbb{R}^n, \varphi' \in \mathbb{R}, \tau' \in \mathbb{R}_+$ such that
3a  $\tau' \geq \lambda, \ x' \in \mathcal{F}$
3b  $f(x') \leq \varphi' \leq \varphi - \sigma(\tau')$
3c  $f(x' + \tau' d) > \varphi' - \sigma(\tau')$ for all $d \in D'$
4a  Keep the best iterate in the global variable $(x_b, f(x_b))$
    RETURN$[x', \varphi', \tau', D']$
```

Algorithm 3 ITERATION() is the core algorithm used in optimization problems with continuous variables in $\mathbb{R}^n$. For unconstrained problems $\mathcal{F} = \mathbb{R}^n$.

In a practical implementation we admit that there is is no qdd at x_{i-1} when (15) holds for $\lambda > \varepsilon, \ \lambda \in \Lambda$ and a finite D_i. When ITERATION() finds a feasible qdd, it returns $[x_i, \varphi_i, \tau_i, D_i]$ satisfying

$$x_i \in \mathcal{F} \tag{16a}$$
$$f(x_i) \leq \varphi_i \leq \varphi_{i-1} - \sigma(\tau_i) \tag{16b}$$
$$f(x_i + \tau_i d) > \varphi_i - \sigma(\tau_i) \text{ for all } d \in D_i. \tag{16c}$$

Lemma 3 ensures that (16) can be fulfilled. ITERATION() also keeps the best estimate x_b at its Line 4a.

Remark 1 φ_i may be the observed value of a random variable uniformly distributed in $\left[f(x_i), \ \varphi_{i-1} - \sigma(\tau_i)\right]$.

3 Nonmonotone DSMs for optimization with variables in $\mathbb{R}^n$

This section applies the theory developed in Sect. 2 to our problems of interest. Section 3.1 deals with the unconstrained optimization problem

$$\underset{x \in \mathcal{F}}{\text{minimize}} \ f(x), \qquad \mathcal{F} = \mathbb{R}^n. \tag{17}$$

This section describes algorithms to find non-smooth stationary points on a finite set of directions. Section 3.2 goes one step forward. It is concerned with convergence

```
        [ $\overset{*}{x}$, $\overset{*}{D}$ ] = CONTINUOUS($x_0, \varphi_0, \varepsilon$)
                minimize $f(x), x \in \mathcal{F}$
        Given $x_0 \in \mathcal{F},\ \varphi_0 \ge f(x_0),\ \varepsilon \ge 0$
.................................................................
        $i = 0$, notdone
        WHILE notdone
            Choose $\tau_i > \varepsilon$
            WHILE ($\tau_i > \varepsilon$)
                $i = i + 1$      (next iteration)
1               $[x_i, \varphi_i, \tau_i, D_i]$ = ITERATION ($x_{i-1}, \varphi_{i-1}$)
            END- WHILE
2a          IF ($f(x_b) < f(x_i)$)   ($x_b, f(x_b)$) is global
2c              $[x_{i-1}, \varphi_{i-1}] = [x_b, f(x_b)]$
            ELSE done
            END- IF
        END- WHILE
3       RETURN $[x_i, D_i]$
```

Algorithm 4 CONTINUOUS() is a non-monotone DSM that returns a stationary point of an optimization problem with variables in $\mathbb{R}^n$. The best iterate $(x_b, f(x_b))$ can be retrieved.

to total stationary points characterized by Definition (6). Previous works have dealt with this issue. To ensure convergence, the authors in [6] need to proceed iteratively with an asymptotically dense matrix. We describe an algorithm which does not require to *explicitly* deal with an asymptotically dense matrix. Section 3.3 deals with the box-constrained optimization problem

$$\underset{x \in \mathcal{F}}{\text{minimize}}\ f_C(x), \qquad \mathcal{F} = \{x \in \mathbb{R}^n : s^k \le x^k \le t^k, k = 1, \ldots, n\}. \tag{18}$$

3.1 Unconstrained optimization

In this section we prove several convergence results for unconstrained problems. Under Assumptions **A1, A2** convergence to a point fulfilling the classical first order necessary conditions is proved. If **A1, A3** hold, a sequence of estimates $\{x_i\}_{i \in \mathbb{N}}$ converges to a stationary point fulfilling Definition 5 for $D = \overset{*}{D}$, a limit point of $\{D_i\}$. By including some more laborious algorithmic implementation in the limit we prove stationarity on the set $\overset{\infty}{D} = \{d : ||d|| = 1\}$.

Our first task is to impose conditions on the searching sets $\{D_i\}$ that ensure convergence to stationary points of smooth functions.

Theorem 1 *Let* $\{d_i\}$ *be the sequence of search directions that satisfies* (12)*, that is:* $d_i^T \nabla f(x_i) \le -\alpha ||\nabla f(x_i)||$), *for some* $\alpha > 0$.
If **A1 and A2** *hold, Algorithm* 4 *generates* $\{\nabla f(x_i)\} \to 0$.

Proof From (16c) and (12) it follows that

$$\begin{aligned} -\alpha\|\nabla f(x_i)\| \geq d_i^T \nabla f(x_i) &= \frac{f(x_i+\tau_i d_i) - f(x_i)}{\tau_i} - o(\tau_i)/\tau_i \\ &\geq \frac{f(x_i+\tau_i d_i) - \varphi_i}{\tau_i} - o(\tau_i)/\tau_i \\ &> -\sigma(\tau_i)/\tau_i - o(\tau_i)/\tau_i. \end{aligned} \tag{19}$$

Since $\tau_i \to 0$, we conclude that $\{\nabla f(x_i)\} \to 0$. □

We claim that the following proposition is valid.

Proposition 1 *Let $\{D_i\}$ be a sequence of search direction sets, each containing a direction of descent satisfying* (12)*. Under Assumptions* **A1** *and* **A2***, Algorithm* 4 *applied to* (17) *generates* $\{\nabla f(x_i)\} \to 0$. □

Corollary 3 *If $f(\cdot)$ is continuously differentiable, any accumulation point $\overset{*}{x}$ of $\{x_i\}$ is stationary; more specifically, $\nabla f(\overset{*}{x}) = 0$.*

Proof It is obvious. □

In general, there is not a simple way to *explicitly* find a qdd at x; however, a qdd might appear in D when:

- $f(\cdot) \in C^2$ and there are directions of negative curvature at x, that is, $d^T \nabla^2 f(x) d < 0$.
- There are directions with a negative directional derivative at x.

We should recall that even if $f(\cdot)$ is differentiable, its first order derivatives might not be computable; and there is no way to explicitly verify (12). Nonetheless, Lemma 4 shows that (12) holds when D_i is a set of orthogonal directions. Computability of the gradient simplifies the algorithm significantly. Convergence prevails for any strictly positive definite matrix P and $D_i = \{-P\nabla f(x_i)\}$, a singleton.

Algorithm CONTINUOUS(), identified as Algorithm 4, is a non-monotone DSM that returns a stationary point to problem (17) regardless of the presence *-or absence-* of derivative information. It is invoked by

$$[\overset{*}{x}, \overset{*}{D}] = \text{CONTINUOUS}(x, \varphi, \varepsilon).$$

CONTINUOUS() calls Algorithm 3 while $\tau_i > \varepsilon$. It has 3 input arguments: the starting point $x_0 \in \mathcal{F}$, the upper function value $\varphi_0 \geq f(x_0)$ and the accuracy ε. CONTINUOUS() finds an nss point when $\tau_i < \varepsilon$ at its Line 1; but it returns x_i only if $f(x_i) < f(x_b)$ at its Line 21. Otherwise, the algorithm invokes ITERATION(x_b, $f(x_b)$). The input parameters $(x_b, f(x_b))$ make sure that all subsequent estimates have a function value below $f(x_b)$. The convergence theory will require $\{x_i\} \subset \mathcal{F}$, which is obvious in unconstrained minimization.

Theorem 2 *Let* **A1, A3** *hold. When $\varepsilon = 0$, Algorithm* 4 *generates $\{\tau_i\} \to 0$, and we can identify a subsequence $\{x_i, D_i, \tau_i\}_{i \in K \subseteq \mathbb{N}}$ converging to $(\overset{*}{x}, \overset{*}{D}, 0)$, where $\overset{*}{x}$ satisfies* (8) *on $\overset{*}{D}$. In addition, $f^0(\overset{*}{x}, d) \geq 0$ for all $d \in \overset{*}{D}$.*

Proof Line 1 of Algorithm 4 invokes ITERATION(). If x_i has no qdds, the IF clause in Line 2b of ITERATION() never holds and $\{\lambda\} \downarrow 0$ within the while loop 2a-2c, but keeping fixed the solution estimate; that is, $x_{j+1} = x_j$ for all $j \geq i$. By compactness we can identify a subsequence $\{x_i, D_i, \tau_i\}_{i \in K \subseteq \mathbb{N}} \to (x_i, \overset{*}{D}, 0)$.
If the whole sequence $\{x_i\}_{i \in \mathbb{N}}$ possess qdds, it follows by (16b) and **A1** that

$$\min_{x \in \mathcal{F}} f(x) \leq f(x_{k+1}) \leq \varphi_{k+1} \leq \varphi_i - \sum_{k \geq j \geq i} \sigma(\tau_j); \tag{20}$$

which implies $\{\sigma(\tau_i)\} \to 0$. A fortiori, $\{\tau_i\} \to 0$. Since $f(x_i) \leq \varphi_i$, it follows from (16c) that

$$\frac{f(x_i + \tau_i d) - f(x_i)}{\tau_i} \geq \frac{f(x_i + \tau_i d) - \varphi_i}{\tau_i} > -\sigma(\tau_i)/\tau_i, \text{ for all } d \in D_i. \tag{21}$$

By compactness we can identify a sequence $\{x_i, D_i, \tau_i\} \to (\overset{*}{x}, \overset{*}{D}, 0)$. Moreover, when **A3** holds, it follows that $f(x_i + \tau_i \overset{*}{d}) \geq f(x_i + \tau_i d) - \vartheta \tau ||d- \overset{*}{d}\,||$, and

$$\frac{f(x_i + \tau_i \overset{*}{d}) - f(x_i)}{\tau_i} \geq \frac{f(x_i + \tau_i d) - \varphi_i}{\tau_i} - \vartheta ||d- \overset{*}{d}\,|| \tag{22a}$$

$$> -\sigma(\tau_i)/\tau_i - \vartheta ||d- \overset{*}{d}\,|| \text{ for all } \overset{*}{d} \in \overset{*}{D}\,. \tag{22b}$$

We conclude that $\overset{*}{x}$ is stationary on the limit set $\overset{*}{D}$. Clearly

$$\limsup_{x \to \overset{*}{x}, \tau \to 0} \frac{f(x + \tau d) - f(x)}{\tau} \geq 0. \tag{22c}$$

□

We just proved that Algorithm 4 generates a sequence $\{x_i\}$ converging to a stationary point $\overset{*}{x}$ satisfying Definition 5 on a limit set $\overset{*}{D}$. We now include material that will guide us to sketch a method to find stationarity on some $D \supset \overset{*}{D}$. We will show stationarity on finite sets $\{D_i\} \to \overset{\infty}{D}$.

3.2 Non-smooth stationarity on $\overset{\infty}{D}$

Remark 2 If (6) holds for the sets $D_1 \subset \overset{\infty}{D}$ and $D_2 \subset \overset{\infty}{D}$, then it also holds for the set $(D_1 \cup D_2)$.

As stated earlier, it is more plausible to implement the algorithm with discrete values for λ.

Remark 3 Let $\hat{\Lambda}$ be given complying with Definition 4, and let x_i be a non-smooth stationary point satisfying (8) on the sets $D_1 \subset \overset{\infty}{D}$ and $D_2 \subset \overset{\infty}{D}$. The point x_i is

```
            $\left[\lambda, d, \#S\right] = \textsc{Total}(y, D, \Lambda)$
  $y$ satisfies (8) on $D$, $\Lambda = \{\lambda_0, \lambda_1, \dots, \}$, $j < k \Rightarrow \lambda_j > \lambda_k$, $\{\lambda_j\} \downarrow 0$
  Find a qdd $d \in S$, a set of random directions unif. dist. in $\overset{\infty}{D}$
.............................................................................
        $S = \{d_1, \dots, d_q\}$ randomly taken from $\overset{\infty}{D}$
        FOR $\lambda \in \Lambda$
          FOR $d \in S$
2a.         IF $\left(y + \lambda d \in \mathcal{F}\right.$ AND $\left. f(y + \lambda d) - f(y) \leq -\sigma(\lambda)\right)$
3a.           RETURN $[\lambda, d, 0]$
            END- IF
          END- FOR
        END- FOR
3b.     RETURN $[0 \in \mathbb{R}, 0 \in \mathbb{R}^n, \#S]$
```

Algorithm 5 The input parameter $y \in \mathbb{R}^n$ is non-smooth stationary satisfying (8) on the set of directions D. TOTAL() determines if y is also non-smooth stationary on a random set of directions S uniformly distributed on $\overset{\infty}{D}$.

also non-smooth stationary satisfying (8) on the set $(D_1 \cup D_2)$. In particular; if x_i is non-smooth stationary on a set $D \subset \overset{\infty}{D}$ and on a set S of random directions uniformly distributed in $\overset{\infty}{D}$, then x_i is also non-smooth stationary on $(D \cup S)$. For the remainder of this sub-section $\{S_i\}$ denotes finite sets of random directions uniformly distributed in $\overset{\infty}{D} = \{d \in \mathbb{R}^n : ||d|| = 1\}$.

Lemma 7 *Let Algorithm 5 be invoked by $\left[\lambda, d, \#S\right] = Total(y, D, \Lambda)$; where $y \in \mathbb{R}^n$ satisfies (8) on (D, Λ). The input y satisfies (8) on the set $D \cup S$ if and only if $\lambda = 0, d = 0$.*

Proof It is elementary. Line 3a of Algorithm 5 returns non-zero values to $\left[\lambda, d\right]$ if -and only if- Line 2a holds true, that is, d is a qdd. □

We close this section with a brief description of the Algorithm 6, which improves the nss point found by Algorithm 5. Theoretically, $\{x_i\}$ converges with probability 1 to an nss point complying with Definition 6.

Remark 4 A uniformly random set of q directions in $\overset{\infty}{D}$ can be efficiently obtained as follows:

rand(0,1) is a random variable with standard normal distribution

```
FOR $j = 1, \dots, q$
  FOR $k = 1, \dots, n$
  $d_j^k = \textbf{rand}(0, 1)$
  END- FOR
  norm $= \sqrt{(d^1)^2 \dots (d^n)^2}$
  IF $(norm > \varepsilon)$  $d \leftarrow d/\text{norm}$
END- FOR
```

In [32] the author suggested that

$$P_r\left(\exists_{i,j,i<j} |d_i^k - d_j^k| < \epsilon\right) < \delta \approx 1 - (1 - \epsilon)^{p(p-1)}, \tag{23}$$

```
$[x, noqdd] = $ NON- SMOOTH$(x, \varepsilon, \Lambda,$ budget$)$
Given the starting point $y \in \mathcal{F}$, accuracies $\varepsilon$, a $\Lambda$-set, a positive budget
$\Lambda = \{\lambda_0, \lambda_1, \ldots, \}, j < k \Rightarrow \lambda_j > \lambda_k, \{\lambda_j\} \downarrow 0$
Find $x$, a non-smooth stationary point
.......................................................................
1.  WHILE (budget $> 0$)
2.      $[x, D] = $ CONTINUOUS$(x, \varphi, \varepsilon)$
        $noqdd = \#D$
        REPEAT
3.          $[\lambda, d, \#S] = $ TOTAL$(x, D, \Lambda)$
            IF $(\lambda = 0)$   $noqdd = noqdd + \#S$
        UNTIL $(\lambda > 0)$ OR (budget $= 0$)
4.      $x \leftarrow x + \lambda d$
    END- WHILE
5.  RETURN $[x, noqdd]$
```

Algorithm 6 Starting point $x \in \mathcal{F}$. The algorithm finds an nss within an estimated budget. The RETURN value is also an nss point on a random set S of directions uniformly distributed on $\overset{\infty}{D}$. $\#S$ depends on the initial budget.

SYNOPSIS:

Line 1: The variable budget expresses the effort needed to identify a large number of no qdds.
Line 2: x is an nss point satisfying (8). It is the first trial solution to the problem.
Line 3: When $\lambda > 0$ a meliorated solution has been generated on a random set S.
Line 4: Update the solution estimate.
Line 5: Return the best estimate.

where $P_r(E)$ is the probability of the event E and p is the number of generated directions.

3.3 Box-constrained optimization

In this subsection we deal with the box-constrained optimization problem (18), which is repeated here for easy reference:

$$\underset{x \in \mathcal{F}}{\text{minimize}}\ f_C(x), \qquad \mathcal{F} = \{x \in \mathbb{R}^n : s^k \le x^k \le t^k, k = 1, \ldots, n\},$$

where s^k, t^k are, respectively, the lower and the upper bound of the variable x^k. We assume $s^k < t^k$; otherwise, when $s^k = t^k$, the variable x^k has a constant value and it is no longer considered as a variable. An approach for solving (18) is to consider the barrier function

$$f(x) = \begin{cases} f_C(x), & \text{if } x \in \mathcal{F}, \\ \varphi_0 + 1, & \text{otherwise,} \end{cases} \tag{24}$$

and try to use the same tools used for solving the unconstrained problem (17). The algorithm starts at $x_0 \in \mathcal{F}$ and applies Algorithm 4 to the function defined by (24). The algorithm also needs that search directions conform to the geometry of the constrained set. Algorithm 2 takes care of this circumstance. To formalize the proof of convergence we need to introduce the following notation and definitions.

– Define the index set B_i of binding variables at the i-th iteration as:

$$B_i = \{1 \leq k \leq n : \min(x_i^k - s^k, t^k - x_i^k) \leq \beta\}, \text{ for some } \beta > 0. \tag{25}$$

– Denote by $\mathcal{S}_i$ the subspace spanned by $\{x \in \mathbb{R}^n : x^k = 0, k \in B_i\}$, and let $v_i \in \mathbb{R}^n$ be the projected gradient on $\mathcal{S}_i$ given by $v_i^k = \begin{cases} \nabla f(x_i)^k, \; k \notin B_i, \\ \quad 0 \qquad \textit{otherwise}. \end{cases}$

– Let $E_i = \{d_1, \cdots, d_q\}$ be a finite set of q vectors in $\mathcal{S}_i$ satisfying (12), that is,

$$\big(\forall v \in \mathbb{R}^n\big)(\exists d \in E_i) : d^T v \leq -\alpha||v||, \text{ for some } \alpha > 0. \tag{26}$$

– Let the set D_i of search directions be

$$D_i = E_i \cup \{\pm e_j, j \in B_i\}. \tag{27}$$

Essentially, we apply Algorithm 4 slightly modified to deal with the binding variables appointed to by B_i. We now proceed to prove convergence.

Let B' be a set that appears infinitely often in the sequence $\{B_i\}_{i \in \mathbb{N}}$.

Let $K' = \{i : B_i = B'\}$ and let $\overset{*}{x}$ be a limit point, that is, $\{x_i\}_{i \in K} \to \overset{*}{x}$, for some $K \subseteq K'$.

Theorem 3 *Let $\{D_i\}$ be the sequence of search directions satisfying* (27). *Under assumptions* **A1, A2**, *Algorithm* 4 *generates a sequence $\{x_i\}_{i \in K}$ converging to a stationary point $\overset{*}{x}$ satisfying:*

$$\begin{aligned} &\big(s^k = \overset{*}{x}{}^k\big) \Rightarrow \nabla f(\overset{*}{x})^k \geq 0, \\ &\big(\overset{*}{x}{}^k = t^k\big) \Rightarrow \nabla f(\overset{*}{x})^k \leq 0, \; \textit{and} \end{aligned} \tag{28a}$$

$$\big(s^k < \overset{*}{x}{}^k < t^k\big) \Rightarrow \nabla f(\overset{*}{x})^k = 0. \tag{28b}$$

Proof Let $B_i, K, \overset{*}{x}$ be defined as above. To prove (28a) we merely prove $\big(s^k = \overset{*}{x}{}^k\big) \Rightarrow \nabla f(\overset{*}{x})^k \geq 0$. The case $\big(\overset{*}{x}{}^k = t^k\big) \Rightarrow \nabla f(\overset{*}{x})^k \leq 0$ is quite similar and it is omitted. As $x_i^k \to \overset{*}{x}{}^k$, it follows that $x_i^k - s^k \leq \beta$ for all large enough i; ergo $k \in B_i$ and $f(x_i + \lambda_i e_k) - f(x_i) \geq -\lambda_i \sigma(\lambda_i)$; e^k satisfies (6). By taking limits we obtain $\nabla f(\overset{*}{x})^k \geq 0$.

To prove (28b) we need to consider 2 cases:

a. $k \in B_i$.
By construction $f(x_i + \lambda_i e_k) - f(x_i) \geq -\lambda_i \sigma(\lambda_i)$ and $f(x_i - \lambda_i e_k) - f(x_i) \geq -\lambda_i \sigma(\lambda_i)$. Together these 2 inequalities imply that $\nabla f(\overset{*}{x})^k = 0$. Besides, both $e^k, -e^k$ satisfy (6).

b. $k \notin B_i$.
By construction E_i is a finite set that contains a direction that satisfies (12). If **A1** and **A2** hold, we mimic the convergence of Theorem 1 *replacing* D by E and $\nabla f(\cdot)$ by v and deduce that Algorithm 4 generates
$\{v_i\}_{i \in K} \to 0$; which means that $\nabla f(\overset{*}{x})^k = 0, k \notin B_i$. □

4 The integer optimization problem

This section is concerned with the integer optimization problem

$$\underset{z\in\mathcal{F}}{\text{minimize}}\, f(z), \qquad \mathcal{F}=\{z\in\mathbb{Z}^p : s\le z\le t\}, \tag{29}$$

where $f(\cdot):\mathbb{R}^p\to\mathbb{R}$. Problem (29) is an optimization problem with integer variables subjected to bounds. It is combinatorial. An exhaustive enumeration of the feasible points might be computationally expensive. Besides, (29) is generally a nonconvex problem and normally no algorithm ensures convergence to the optimizer. To try to find a global solution several strategies based on relaxations, cutting planes, branch and bound, surrogate models and *heuristics* have been devised. It seems out of the question to elaborate a common approach for getting the global solution to any instance of (29). However, some attempts have solved specific instances. In [7], the authors adapt MADS for solving a problem with *grid* variables regularly distributed along the coordinate axis. In [19], the authors applied a discretized version of Algorithms 3 and 4 for solving the same problem. We recall that this paper assumes that constraints other than bounds can be handled via penalization [12, 33], Lagrangian [30], infinite barriers [1] or any other appropriate technique.

Our algorithm is monotone with no σ-function involved. From the onset, we admit that we have at hand a feasible starting point $z_0\in\mathcal{F}$ and only feasible points are considered as solution estimates. We also assume that **A1** holds, which essentially means that the level set $L=\{z\in\mathcal{F} : f(z)\le f(z_0)\}$ is finite. We consider convergence to local minimizers. A *local* neighborhood $\mathcal{N}(z_i,\varrho)$ can be defined as

$$\mathcal{N}(z_i,\varrho)=\{v\in\mathcal{F} : ||v-z_i||\le\varrho\}, \tag{30}$$

where $||\cdot||$ stands for any norm and $\varrho>0$. Usually the iterate z_i is considered stationary to (29) if

$$v\in\mathcal{N}(z_i,\varrho)\Rightarrow f(v)\ge f(z_i). \tag{31a}$$

A naive artifice to verify (31a) constructs $\mathcal{N}(z_i,\varrho)$ and then evaluates all $v\in\mathcal{N}(z_i,\varrho)$. This is often an expensive procedure that precludes the use of $\varrho>2$. The algorithm Greedy(), identified as Algorithm 7, is described above. It tries to find better estimates that do not belong to the set $\mathcal{N}(z_i,\varrho)$; z_i will be accepted as stationary if

$$v\in\mathcal{V}(z_i)\Rightarrow f(v)\ge f(z_i), \tag{31b}$$

where $\mathcal{V}(z_i)\supseteq\mathcal{N}(z_i,\varrho)$. The aim is to build $\mathcal{V}(z_i)-\mathcal{N}(z_i,\varrho)\ne\emptyset$. This can be regarded as a strategy to try to escape from the local stationary point defined by (31a).

We now briefly illustrate the way Greedy() constructs $\mathcal{V}(z_i)$.

Given $q\in\{1,\ldots,p\}$ we want

$$\mathcal{V}_q(z_i)\subseteq\big\{v\in\mathcal{F} : ||v-z_i||_0=q\big\}. \tag{32a}$$

```
                         $[y] = \text{GREEDY}(z, \eta)$
        Given $z \in \mathcal{F}$, $\eta \in \{1, 2, \ldots, p\}$, finds $y$ satisfying (33)
             minimize $f(z), z \in \mathcal{F} = \{z \in \mathbb{Z}^p : s \leq z \leq t\}$
.......................................................................................
 1.   $q = \eta$, $y = z$
 2.   WHILE $q > 0$
3a.     IF $(q > 1)$
3b.       $u = 0$, Pick $q$ random indices,                  $j = i1, \ldots, iq$
3c.       $u^j \in \{-2, -1, 1, 2\}$                          $j = i1, \ldots, iq$
3d.       $\tau = u^T u$
3e.       $d_j = e_j - (2u^j/\tau)u$,                         $j = i1, \ldots, iq$
3f.     ELSE
3g.       $\tau = 1, d_j \in \{d : d = v - y, v \in \mathcal{N}(y, \varrho)\}$,   $j = 1, \ldots, \#\mathcal{N}(y, \varrho)$
3h.     END- IFELSE
4a.     $\mathcal{V} = \{v \in \mathcal{F} : v = y \pm \tau d_j\}$
4b.     IF $\big(\exists\, (v' \in \mathcal{V}) : f(v') < f(y)\,\big)$
4c.       $y = v'$,   $q = \eta$
4d.     ELSE $(q = q - 1)$
4e.     END- IFELSE
 5.   END- WHILE
 6.   RETURN $[y]$
```

Algorithm 7 GREEDY() algorithm for solving (29), an optimization problem with integer variables. The output y is a stationary point satisfying (33).

SYNOPSIS:

Line 1:	Assigns $q = \eta$. Saves z as a possible stationary point, GREEDY() looks for a better estimate in $\mathcal{V}_q \subseteq \{z \in \mathbb{Z}^p : \|\|z - y\|\|_0 \leq q\}$.
Line 3b:	Picks randomly q components $i1, \ldots, iq$.
Lines 3b-3f:	Randomly generates orthogonal search directions $d_{i1}, \ldots, d_{iq}$ with q components each.
Line 3g:	This line is only executed when $q = 1$. Its purpose is to ensure that all points in $\mathcal{N}(y, \varrho)$ are evaluated.
Line 4a:	Defines the set $\mathcal{V}_q$ with $2q$ members.
Lines 4b-4e:	If (y does not satisfy (33)) the algorithm repeats the WHILE loop with $y = v', q = h$. This means that v' is now the candidate for being the stationary point.
Line 4d:	Otherwise, q is reduced.

With no loss of generality we assume $v^k = z_i^k$, for $k > q$ and rewrite (32a) as:

$$\mathcal{V}_q(z_i) \subseteq \big\{v \in \mathcal{F} : v^k = z_i^k, k > q\big\}. \tag{32b}$$

In the actual implementation $v^{k_1} = z_i^{k_1}, \cdots, v^{k_q} = z_i^{k_q}$, with $k_1, \ldots, k_q$ randomly chosen. The generality of (32b) is preserved, but the explanation that follows is more fluid. We make use of search directions and rewrite (32b) as

$$\mathcal{V}_q(z_i) \subseteq \big\{v \in \mathcal{F} : v = z_i + \tau d, d^k = 0, k > q\big\}. \tag{32c}$$

We have not yet defined the set $\mathcal{V}_q(z_i)$, but a promising candidate is:

$$\mathcal{V}_q(z_i) = \big\{v \in \mathcal{F} : v = z_i + (u^T u)He^k, k \leq q\big\}, \tag{32d}$$

where $H = I - (2/u^T u)uu^T$ is the $p_\times p$ Householder matrix, $u^k \in \{-2, -1, 1, 2\}$ for $k \leq q$, and $u^k = 0, k > q$.

The set $\mathcal{V}_q(z_i)$ is the set E generated by Algorithm 2 specialized to discrete variables. Given $h \leq p$, y is accepted as stationary if

$$f(y) \leq f(v), v \in \mathcal{V}(y) = \bigcup_{q=1}^{h} \mathcal{V}_q(y). \tag{33}$$

Remark 5 In our implementation we define $\mathcal{V}(y) = \bigcup_{q=1}^{h} \mathcal{V}_q(y) \cup \mathcal{N}(y, \varrho)$.

GREEDY() is invoked by $y =$ GREEDY (z_i, η), where y is stationary satisfying (33), z_i is the variable under scrutiny, and $||y - z_i||_0 \leq \eta$.

This algorithmic implementation was effective on the numerical tests performed in Sect. 6. A global solution was often returned. A detailed algorithmic description of GREEDY() is given just after its pseudocode is displayed in Algorithm 7.

Remark 6 With our implementation $||z_i - z_{i-1}||_0 = p$ is possible.

Lemma 8 *If* **A1** *holds,* GREEDY() *returns a stationary point satisfying* (33) *in a finite number of iterations.*

Proof If GREEDY() remains in the WHILE loop, the IF clause in Line 4b is valid infinitely often and Line 4c generates an infinite strictly decreasing sequence $\{f(y)\}$. By **A1** this cannot occur; hence, GREEDY() returns with $q = 0$. When $q = 1$, GREEDY()'s Line 3g constructs the search directions $d_1, \ldots, d_{\#\mathcal{N}}$ and reset $\mathcal{V}$ to $\mathcal{N}(z, \varrho)$. Hence, $q = 0$ only happens if $f(v) \geq f(z)$, for all $v \in \bigcup_{q=1}^{h} \mathcal{V}_q(z) \cup \mathcal{N}(z, \varrho)$. □

5 Mixed integer optimization

In this section we analyze convergence of the Variable Separation (VS) scheme for solving problem (1), which is repeated here for easy reference.

$$\underset{(x;z)\in\mathcal{F}}{\text{minimize}} \; g(x; z) : \mathbb{R}^{n+p} \to \mathbb{R},$$
$$\mathcal{F} = \{x \in \mathbb{R}^n, \; z \in \mathbb{Z}^p : s \leq (x; z) \leq t\}.$$

The difficulties, the standard methodology and the software that has been used for solving mixed integer optimization problems is abridged in [8, 28]. The authors in [19] suggested a uniform discretization of the continuous variables on a grid and solve (29). To improve the solution, the problem is again solved on a finer grid until convergence criteria are met. Details can be found in [19].

5.1 Neighborhood

The neighborhood definition is crucial to detect good optimizers. An algorithm that returns $\overset{*}{z}$ as a solution to a discrete problem must ensure that no better point is found in its neighborhood. A recent discussion on this issue was given in [36] in the context of DFO. We extract 2 definitions of local stationarity depending on a specified neighborhood. $(\overset{*}{x}; \overset{*}{z}) \in \mathcal{F}$ is a local minimizer of a nonlinearly constrained problem with mixed variables if either **A** or **B** hold.

$$\mathbf{A}: \quad g(\overset{*}{x}; \overset{*}{z}) \le g(x; z) \text{ for all } x \in \mathcal{B}(\overset{*}{x}, \overset{*}{z}, \rho) \text{ and all } z \in \mathcal{N}(\overset{*}{x}, \overset{*}{z}, \varrho), \tag{34a}$$

$$\text{where } \mathcal{B}(\overset{*}{x}, \overset{*}{z}, \rho) = \{x : (x; \overset{*}{z}) \in \mathcal{F}, ||x - \overset{*}{x}|| \le \rho\}, \tag{34b}$$

$$\text{and } \mathcal{N}(\overset{*}{x}, \overset{*}{z}, \varrho) = \{z : (\overset{*}{x}; z) \in \mathcal{F}, ||z - \overset{*}{z}|| \le \varrho\}. \tag{34c}$$

$$\mathbf{B}: \quad g(\overset{*}{x}; \overset{*}{z}) \le g(x; z) \text{ for all } x \in \mathcal{B}'(\overset{*}{x}, \rho) \text{ and all } z \in \mathcal{N}'(\overset{*}{z}, \varrho), \tag{35a}$$

$$\text{where } \mathcal{B}'(\overset{*}{x}, \rho) = \{x : (x; z) \in \mathcal{F} \text{ for some } z, ||x - \overset{*}{x}|| \le \rho\}, \tag{35b}$$

$$\text{and } \mathcal{N}'(\overset{*}{z}, \varrho) = \{z : (x; z) \in \mathcal{F} \text{ for some } x, ||z - \overset{*}{z}|| \le \varrho\}. \tag{35c}$$

It is easy to see that $\mathcal{B}(\overset{*}{x}, \overset{*}{z}, \rho) \subseteq \mathcal{B}'(\overset{*}{x}, \rho)$, and $\mathcal{N}(\overset{*}{x}, \overset{*}{z}, \varrho) \subseteq \mathcal{N}'(\overset{*}{z}, \varrho)$. The authors in [36] claim some benefits from the use of **B**.

Definition 7 The point $(\overset{*}{x}; \overset{*}{z})$ is locally stationary for problem (1) if

$$\min_{z \in \mathcal{V}(\overset{*}{z})} g(\overset{*}{x}; z) = g(\overset{*}{x}; \overset{*}{z}) = \min_{x \in \mathcal{B}(\overset{*}{x}, \overset{*}{z}, \rho)} g(x; \overset{*}{z}), \tag{36}$$

where we are employing the neighborhood $\mathcal{V}(z_i)$ given by (32) in the previous section. Notationally $\mathcal{V}(z_i) \subseteq \{z : (x_i, z) \in \mathcal{F}\}$.

5.2 Variable separation for solving (1)

The standard approach for solving (1) is based on the VS scheme. VS is a powerful technique that has been used in optimization problems in a multi-processing environment. It is specially suitable for solving large systems [1, 31]. Convergence for DFO with VS and continuous variables was first established in [3] for monotone algorithms and later for nonmonotone DSMs [16].

A typical VS scheme splits a problem into two or more subproblems. In mixed integer problems, it seems natural to divide the problem (1) in the subproblems (37a, 37b). Each can be worked out with specific tools. In the rest of the paper, we commit a minor abuse of notation and $\mathcal{B}(x_{i-1}, \rho)$ substitutes for $\mathcal{B}(x_{i-1}, z_{i-1}, \rho)$.

$$\text{Given } (x_{i-1}; z_{i-1}) \in \mathcal{F}$$

$$\text{find } x_i \text{ such that } g(x_i; z_{i-1}) \le g(x; z_{i-1}) \text{ for } x \in \mathcal{B}(x_{i-1}, \rho) \tag{37a}$$

$$\text{and}$$

$$\text{find } z_i \text{ such that } g(x_i; z_i) \le g(x_i; z) \text{ for } z \in \mathcal{V}(z_{i-1}). \tag{37b}$$

Given $(x_{i-1}; z_{i-1}) \in \mathcal{F}$, (37a) gives x_i, the best local solution for x provided that $z_{i-1} \in \mathbb{Z}^p$ stays fixed, then (37b) selects the best discrete variable in $\mathcal{V}(z_{i-1})$, a finite set of feasible points that includes z_{i-1}. This iterative process is repeated until convergence conditions are met. In [36] the authors suggest to look for a global solution to (37b) and claim some benefits [36, Section 2]. However, their choice may be computationally expensive, since it looks for the global optimum of a problem with discrete variables. A local solution will hopefully require the evaluation of $g(x_i; z)$ at points in a small discrete set. In some applications the subproblem (37a) reduces to a linear model and z_i is chosen with heuristics linked to the structure of the problem [39]. It is advisable to use known efficient techniques whenever possible.

```
$[\overset{*}{x}; \overset{*}{z}] = $ STANDARDVS$(x_0, z_0, \varepsilon)$
.................................................................
        Input: $(x_0; z_0) \in \mathcal{F},\ \varepsilon > 0$
        Output: A stationary point to Problem (1)
.................................................................
$i = 0$
DO
    $i = i + 1$
    Find $x_i$ such that:
        $g(x_i; z_{i-1}) = \min_{x \in \mathcal{B}(x_{i-1}, \rho)} g(x; z_{i-1})$        (38a)
    Compute $z_i$ such that:
        $g(x_i; z_i) = \min_{z \in \mathcal{V}(z_{i-1})} g(x_i; z)$        (38b)
WHILE $\big(g(x_{i-1}; z_{i-1}) - g(x_i; z_i) > \varepsilon\big)$        (38c)
RETURN $(x_i; z_i)$
```

Algorithm 8 STANDARDVS() uses $\mathcal{V}(z_{i-1})$ for solving problem (1)

STANDARDVS(), identified as Algorithm 8, describes a straightforward implementation of (37). It is invoked with $[\overset{*}{x}; \overset{*}{z}] =$ STANDARDVS(x_0, z_0, ε), where $(x_0; z_0)$ is the actual estimate, ε is the accuracy, and $(\overset{*}{x}; \overset{*}{z})$ is the stationary point satisfying (36). We now prove convergence under Assumptions **A1, A4**. We assume that we have at hand algorithms that solve problems (38a, b)

A1 The function $g(\cdot; \cdot)$ is computable at all feasible points and the set
$L = \{(x; z) \in \mathcal{F} : g(x; z) \le \varphi_0\}$ is compact for any given φ_0.

A4 The function $g(\cdot; \cdot) : \mathbb{R}^{n+p} \to \mathbb{R}$ satisfies the conditions required by the algorithms employed for solving (38a, b).

Lemma 9 *Under assumptions* **A1, A4** *the sequence* $\{g(x_i; z_i)\}$ *generated by* STANDARDVS() *is decreasing.*

Proof By optimality in (38a) and (38b) it follows that:

$$\min_{z \in \mathcal{V}(z_{i-1})} g(x_i; z) = g(x_i; z_i) \le g(x_i; z_{i-1}), \tag{39a}$$

and

$$g(x_{i-1}; z_{i-1}) \ge g(x_i; z_{i-1}) = \min_{x \in \mathcal{B}(x_{i-1}, \rho)} g(x; z_{i-1}). \tag{39b}$$

Hence,

$$g(x_{i-1}; z_{i-1}) \geq g(x_i; z_{i-1}) \geq g(x_i; z_i) \tag{40}$$

□

Corollary 4 *If $g(x_i; z_i) = g(x_{i-1}; z_{i-1})$, then $g(x_i; z_i)$ satisfies (36).*

Proof If $g(x_i; z_i) = g(x_{i-1}; z_{i-1})$, then from (40) we deduce that $g(x_{i-1}; z_{i-1}) = g(x_i; z_{i-1})$ and (39a-b) becomes a chain of equalities. It follows that $g(x_{i-1}; z_{i-1})$ satisfies (36) and Algorithm 8 stops. □

Lemma 10 *Let ρ be large enough and let $z_j = z_{i-1}$ for some $j > i$. Under Assumptions* **A1, A4** *Algorithm 8 stops at a stationary point.*

Proof At iterations i, j, with $j > i$ we obtain by construction that

$$\begin{array}{c} g(x_{i-1}; z_{i-1}) \geq g(x_i; z_{i-1}) \geq g(x_i; z_i) \\ \textit{and} \\ g(x_j; z_j) \geq g(x_{j+1}; z_j) \geq g(x_{j+1}; z_{j+1}). \end{array} \tag{41}$$

We now assume $z_j = z_{i-1}$ and reach a contradiction. It follows that

$$g(x_{j+1}; z_j) = \min_{x \in \mathcal{B}(x_j, \rho)} g(x; z_j) \tag{42a}$$

$$= \min_{x \in \mathcal{B}(x_j, \rho)} g(x; z_{i-1}) = \min_{x \in \mathcal{B}(x_{i-1}, \rho)} g(x; z_{i-1}) \tag{42b}$$

$$= g(x_i; z_{i-1}), \tag{42c}$$

where equality (42b) holds because $\mathcal{B}(x_j, \rho) = \mathcal{B}(x_{i-1}, \rho)$ for large enough ρ. It also follows that

$$\begin{array}{rll} g(x_i; z_{i-1}) & \geq g(x_i; z_i) & \text{by (41)} \\ & > g(x_j; z_j) & \text{by Corollary 4} \\ & \geq g(x_{j+1}; z_j) & \text{by (42a)} \\ & \geq g(x_i; z_{i-1}) & \text{by (42c);} \end{array} \tag{43}$$

which is a clear contradiction. □

From the above discussion we conclude that

Proposition 2 *Under Assumptions* **A1, A4** *and large enough ρ, Algorithm 8 terminates in a finite number of iterations.* □

Proof It is an immediate consequence of Lemma 10 and Corollary 4. On the one hand, if $z_j = z_{i-1}$ for some $j > i$, Lemma 10 ensures termination in a finite number of iterations. On the other hand, since the feasible set is bounded by **A1**, all integer variables are exhausted in a finite number of iterations. □

This is a desirable conclusion. However, it requires the global solution of optimization problems at each iteration. Theoretically, this is an infinite process. When either optimization problem, (38a) or (38b), is rather complex, we offer another alternative that avoids global optimization subproblems.

5.3 VS with incomplete optimization

If we denote $w = \begin{pmatrix} x \\ z \end{pmatrix} \in \mathbb{R}^{n+p}$, problem (1) can be considered as an optimization problem with $n+p$ variables. To prove convergence along the lines laid out in Sect. 2, we should construct a set of search directions $D = \{d_1, \dots, d_q\}, q > (n+p), d_j \in \mathbb{R}^{n+p}$, with some $d \in D$ satisfying (12). A lot of information is needed. The new estimate is

$$w_i = w_{i-1} + \tau_i d_j, \text{ that is, } \begin{pmatrix} x_i \\ z_i \end{pmatrix} = \begin{pmatrix} x_{i-1} \\ z_{i-1} \end{pmatrix} + \tau_i \begin{pmatrix} d_j^x \\ d_j^z \end{pmatrix} \tag{44}$$

for some $d_j \in D$; d_j^x are the first n components of d_j, and d_j^z stand for the last p components of d_j. This updating might not look convincing. To keep $\{z_i\} \in \mathbb{Z}^p$, we must impose $\{\tau_i d_j^z\} \in \mathbb{Z}^p$. This restriction on $\{\tau_i\}$ is not desirable for the continuous variables and updating (44) is discarded. The VS scheme used by Algorithm 8 overcomes this shortcoming. However, the solution of the global optimization problems (38a, b) are required to prove convergence. In this section we propose a VS scheme that does not require global optimization. It strongly rests upon the theory developed in the previous sections. We need to define stationarity for Problem (1)

Definition 8 $(\overset{*}{x}; \overset{*}{z}) \in \mathcal{F}$ is nonsmooth stationary on D for Problem (1) if $\exists(\{x_i\} \to \overset{*}{x}$ $, \{\tau_i\} \downarrow 0)$ such that

$$x_i + \tau_i d \notin \mathcal{F} \textbf{or} \left(g(x_i + \tau_i d; \overset{*}{z}) - g(x_i; \overset{*}{z}) > -\sigma(\tau_i)\right) \text{ for } d \in D, \tag{45a}$$

and for some $\varrho > 0$

$$g(\overset{*}{x}; \overset{*}{z}) \leq g(\overset{*}{x}, z) \text{ for } z \in \mathcal{V}(\overset{*}{z}, \varrho). \tag{45b}$$

Algorithm 9 is the MAIN() algorithm. It uses a VS scheme for solving Problem (1). Given $(x_{i-1}; z_{i-1})$, the MAIN() algorithm

1. invokes MIXEDDISCRETE(), identified as Algorithm 11, to obtain the next iterate $(x_{i-1}; z_i)$. Its output z_i satisfies

$$\begin{aligned} (x_{i-1}; z_i) \in \mathcal{F},\ g(x_{i-1}; z_i) \leq g(x_{i-1}; z)\} \text{ for } z \in \mathcal{V}(z_{i-1}), \text{ and} \\ g(x_{i-1}; z_i) \leq g(x_{i-1}; z_{i-1}). \end{aligned} \tag{46}$$

The last inequality trivially holds if $z_{i-1} \in \mathcal{V}(z_{i-1})$.

$$[\overset{*}{x};\overset{*}{z}] = \text{MAIN}(x_0, z_0, \eta, \varepsilon)$$

$$\underset{(x;z)\in\mathcal{F}}{\text{minimize}}\, g(x;z) : \mathbb{R}^{n+p} \to \mathbb{R}$$
$$\mathcal{F} = \{(x;z) : x \in \mathbb{R}^n, z \in \mathbb{Z}^p, s \le (x;z) \le t\}$$

Input: $(x_0; z_0) \in \mathcal{F}$, $\eta \in \{1, \dots, p\}$, $\varepsilon > 0$

Output: $(\overset{*}{x};\overset{*}{z})$ satisfying (45)

```
        $i = 0$
1.      DO
          $i = i + 1$        next iteration
2.        $z_i = \text{MIXEDDISCRETE}\left(x_{i-1}, z_{i-1}, \eta\right)$
3a.       $v = x_{i-1}$        auxiliary variable
3b.       REPEAT
3c.          $\left[v,\ \varphi_i, \tau_i, D_i\right] = \text{MIXEDCONTINUOUS}\,(v, z_i, \varphi_{i-1})$
3d.       UNTIL $(||v - x_{i-1}|| \le \varepsilon)$ OR $\left(g(v; z_i) < g(x_{i-1}; z_i)\right)$
4a.       $x_i = v$
        WHILE $||x_i - x_{i-1}|| > \varepsilon)$ OR $||z_i - z_{i-1}||_0 > 0$
        RETURN $(x_i; z_i)$
```

Algorithm 9 MAIN() algorithm for solving the mixed integer optimization problem (1) by Variable Separation in continuous and discrete variables.

$$[x_i, \varphi_i, \tau_i, D_i] = \text{MIXEDCONTINUOUS}(x_{i-1}, z_i, \varphi_{i-1})$$

Input: $(x_{i-1}; z_i) \in \mathcal{F}$, $\varphi_{i-1} \ge g(x_{i-1}; z_i)$

Output: $(x_i; z_i) \in \mathcal{F}$, $\varphi_i \ge g(x_i; z_i)$, $\tau_i \in \mathbb{R}_+$, AND $D_i = \{d_1, \dots, d_q\}$, q directions in $\mathbb{R}^n$

```
1.      Generate a set $E$ with Algorithm 2, and construct $D_i \supseteq E$
2a.     IF $\left(\exists\{\lambda_j\} \to 0\right)$ such that (47) holds
2b.        RETURN $[x_{i-1}, \varphi_{i-1}, 0, D_i]$
        ELSE Find $x_i \in \mathbb{R}^n$, $\varphi_i \in \mathbb{R}$, $\tau_i \in \mathbb{R}$ satisfying
3a.        $(x_i; z_i) \in \mathcal{F}$
3b.        $g(x_i; z_i) \le \varphi_{i-1} - \sigma(\tau_i)$
3c.        $g(x_i; z_i) \le \varphi_i \le \varphi_{i-1} - \sigma(\tau_i)$
3d.        $g(x_i + \tau_i d; z_i) > \varphi_i - \sigma(\tau_i)$ for all $d \in D_i$
3e.        RETURN $[x_i, \varphi_i, \tau_i, D_i]$
4.      END- IFELSE
```

Algorithm 10 (Algorithm MIXEDCONTINUOUS). Section of the MAIN() algorithm to handle the continuous variables x in the mixed integer problem (1).

.

2. invokes MIXEDCONTINUOUS(), identified as Algorithm 10, within a REPEAT- UNTIL loop. The algorithm leaves the loop when $g(x_i; z_i) < g(x_{i-1}; z_i)$. This strategy makes sure the algorithm generates a decreasing sequence; $g(x_i; z_i) > g(x_j; z_j)$ for $i < j$. This feature is shared with Algorithm 8 and it was very convenient for proving finite termination. Besides, if the decreasing condition does nold hold, the algorithm might not be converging to the optimum solution $(\overset{*}{x};\overset{*}{z})$ returned by STANDARDVS().

Remark 7 Algorithms 10 and 11 are, respectively, Algorithms 3 and 7 adapted to the VS scheme for solving Problem (1). They are explicitly shown to facilitate the writing and reading of this section.

```
[$z_i$] = MIXEDDISCRETE($x_{i-1}, z_{i-1}, \eta$)
        $\underset{x=x_{i-1},(x;z)\in\mathcal{F}}{\text{minimize}}\ g(x; z)$
Input: $(x_{i-1}; z_{i-1}) \in \mathcal{F},\ h \in \{1, \dots, p\}$. Output: $(x_{i-1} : z_i) \in \mathcal{F}$
 1.   $q = \eta, z_i = z_{i-1}$
 2.   WHILE  $q > 0$
3a.     Pick $q$ random indices,      $j = i1, \dots, iq$
3b.     IF ($q > 1$)
3c.       $u^j \in \{-2, -1, 1, 2\}$      $j = i1, \dots, iq$
3d.       $\tau = u^T u$
3e.       $d_j = e_j - (2u^j/\tau)u$,      $j = i1, \dots, iq$
3f.     ELSE
3g.       $\tau = 1,\ d_j : z_i + d_j \in \mathcal{N}(z_i, \varrho)$
3h.     END- IFELSE
4a.     $\mathcal{V} = \{(x_{i-1}; v) \in \mathcal{F} : v = z_i \pm \tau d_j\}$
4b.     IF $\big(\exists\ (x_{i-1}; v') \in \mathcal{V} : g(x_{i-1}; v') < g(x_{i-1}; z_i)\big)$
4c.       $z_i = v'$,   $q = \eta$
4d.     ELSE  ($q = q - 1$)
4e.     END- IFELSE
 5.   END- WHILE
 6.   RETURN $z_i$
```

Algorithm 11 (Algorithm MIXEDDISCRETE()). Section of the MAIN() algorithm to handle the discrete variables z in the mixed integer problem (1).

Lemma 11 *Given* $(x_{i-1}; z_{i-1}) \in \mathcal{F}$*, let* z_i = MIXEDDISCRETE(x_{i-1}, z_{i-1}, η). *It follows that* $g(x_{i-1}; z_i) \le g(x_{i-1}; z)$ *for all* $z \in \mathcal{V}(z_{i-1})$.

Proof This is a replica of Lemma 8, replacing $f(z)$ by $g(x; z)$. We reach a similar result, namely, $g(x_{i-1}; z_i) \le g(x_{i-1}; v),\ v \in \bigcup_{q=1}^{h} \mathcal{V}_q(y)$. □

Let x_{i+1} be returned by MIXEDCONTINUOUS$(x_i, z_{i+1}, \varphi_i)$. If a qdd is not found by MIXEDCONTINUOUS(), then $\exists\{\lambda_j\} \downarrow 0$ such that

$$(x_i + \lambda d, z_i) \notin \mathcal{F}, \textbf{ or} \tag{47a}$$

$$g(x_i + \lambda d; z_i) - g(x_i; z_i) > -o(\lambda) \text{ for } \lambda \in \{\lambda_j\}, d \in D_{i+1}. \tag{47b}$$

MAIN() returns $(x_i; z_i)$ as stationary. In a practical implementation we admit that a qdd at x_i does not exist when $\lambda < \varepsilon$ in (47). In short, MAIN() returns (x_i, z_i) satisfying (47) or it generates an infinite sequence $\{x_i, z_i, \tau_i, \varphi_i, g_i, D_i\}$ with the following properties:

$$(x_{i-1}; z_i) \in \mathcal{F}, \tag{48a}$$

$$g(x_{i-1}; z_i) < g(x_{i-1}; z), \text{ for all } z \in \mathcal{V}(z_{i-1}). \tag{48b}$$

$$g(x_{i-1}; z_i) \leq g(x_{i-1}; z_{i-1}), \tag{48c}$$
$$(x_i; z_i) \in \mathcal{F}, \tag{48d}$$
$$g(x_i; z_i) < g(x; z_i) \text{ for all } x \in \mathcal{B}(x_{i-1}, \rho) \tag{48e}$$
$$g(x_i; z_i) \leq \varphi_i \leq \varphi_{i-1} - \sigma(\tau_i), \tag{48f}$$
$$g(x_i + \tau_i d; z_i) > \varphi_i - \sigma(\tau_i) \text{ for all } d \in D. \tag{48g}$$

The following facts are inherited from the material analyzed in the previous sections of this paper. We should bear in mind that $f(x) \overset{\Delta}{=} g(x; \overset{*}{z})$, whenever $\overset{*}{z}$ is fixed.

Lemma 12 *Under Assumption* **A1**, $\{\tau_i\} \to 0$.

Proof By (48e) and **A1** it follows that

$$g(x_k; z_k) \leq \varphi_k \leq \varphi_i - \sum_{k>j\geq i} \sigma(\tau_j). \tag{49}$$

This implies that $\{\sigma(\tau_i)\} \to 0$. A fortiori, $\{\tau_i\} \to 0$. □

Theorem 4 *Under Assumptions* **A1, A4**, $\{x_i\}_{i\in K} \to \overset{*}{x}$ *satisfying* (45).

Proof Let $S = \{x_i, z_i, \tau_i, g_i, \varphi_i, D_i\}_1^\infty$ be the infinite sequence generated by MAIN() and let $\overset{*}{x}, \overset{*}{D}$ be an accumulation point of $\{x_i, D_i\}$. As the number of discrete variables is finite by **A1**, there exists a variable, say z_k, that appears infinitely often in S. Let us denote this variable as $\overset{*}{z}$ and extract from S the subsequence

$$\overset{*}{S} = \{x_i, \overset{*}{z}, \tau_i, g_i, \varphi_i, D_i\}_{i\in K}, K = \{i : \{x_i, D_i\} \to (\overset{*}{x}, \overset{*}{D}), z_i = \overset{*}{z}\}.$$

From (48f) it follows for $i \in K$ that

$$\begin{aligned}\frac{g(x_i + \tau_i d; \overset{*}{z}) - g(x_i; \overset{*}{z})}{\tau_i} &\geq \frac{g(x_i + \tau_i d, \overset{*}{z}) - \varphi_i}{\tau_i} \\ &> -\sigma(\tau_i)/\tau_i, \text{ for all } d \in D_i,\end{aligned} \tag{50}$$

which shows that (47) holds. Furthermore, from Lemma 11 we obtain that $g(x_i; \overset{*}{z}) \leq g(x_i, z),\ z \in \mathcal{V}(\overset{*}{z}, \varrho)$. We conclude that (45) holds. The proof is complete. □

6 Numerical experiments

This section reports some results obtained with a code written in C and compiled with DEV-cpp version 5–11 on a computer equipped with an Intel Core CPU @ 3.3 GHz. These preliminary results emphasize four relevant goals:

1. *Continuous vs Discrete Performance.* All problems tested have global integer optimizers. We ran two algorithms for each problem: CONTINUOUS() with $x \in \mathbb{R}^n$ and GREEDY() with $x \in \mathbb{Z}^n$. We would like to show *numerically* that both algorithms converge adequately to the global solution. We point out that it is common to include the constraint $x \in \mathbb{Z}^n$ to test integer optimization algorithms [37, 38, 43].

2. *Global solution (glob)*. We carried out 100 runs for each problem and report the number of cases where the known global optimum was attained.
3. *Search directions*. The experiments were performed with the randomly generated directions described by Algorithm 2 and with a coordinate search.
4. *Functional evaluations (eval)*. This figure is often considered a performance index.

The choice of parameters may significantly influence any algorithm's behavior. The best selection is probably problem dependent. Nonetheless we run all examples with a common set of parameters,

- $\epsilon = 1.e\text{-}07$ is the required accuracy. The smaller its value, the larger the value of *eval*. We use ϵ mainly to stop the algorithms.
- $\eta = \min(n, 6)$, where n is the number of variables. The η value suggests the size of the local neighborhood where a better stationary point may be located. The larger its value, the larger the value of *eval*.
- $\varphi_0 = f(x_0; z_0) + 0.8|f(x_0; z_0)|$. This parameter affects the non-monotone behavior of the algorithm.
- $\delta = 1.e\text{-}05$. To inhibit stagnation we accept x_{i+1} as a new estimate when:

$$\begin{gathered} f(x_{i+1}) \leq \phi_i - \sigma_i, \\ \textbf{and} \\ |f(x_{i+1}) - f(x_i)| \geq \delta(\delta + |f(x_i)|). \end{gathered} \tag{51}$$

The test problems have a small number of variables; however, they seem to represent a collection of hard problems. The starting point was a random feasible $z \in \mathbb{Z}^n$. Whenever possible, we state some comparison related issues with other algorithms that solve the same problem. The results are summarized in Tables 1, 2, 3, 4, 5, 6, 7 and 8. Each table shows the reference to the problem studied; its number and kind of variables; the set of search direction; the minimum, maximum and average number of function evaluations, and the number of times the known global solution is attained.

The algorithm also solved a small nonlinear real problem described in [10, Section 3.3]. The idea is to show an artifice to deal with discrete variables, which are not integer variables. It is worth mentioning that our algorithm unveiled a better solution than that reported in [10]. This section ends with some remarks about the numerical results obtained.

6.1 Branin function [9]. Optimizer: y= (−3, 13)

$$\begin{aligned} a = 1, b = 5.1/(4\pi^2), c = 5.0/\pi, \\ r = 6.0, s = 10.0, t = 1.0/(8\pi), \\ x^1 = y^1 - 0.689, x^2 = y^2 + 0.629. \end{aligned}$$

$$\begin{gathered} f(x) = a * \left(x^2 - b(x^1)^2 + cx^1 - r\right)^2 + s(1-t)\cos(x^1) + s + 5x^1. \\ -5 \leq x^1 \leq 10; \quad 0 \leq x^2 \leq 15. \end{gathered} \tag{52}$$

The Branin function is a classical test problem in global optimization.

Table 1 Performance for function (52), n = 2

var	Variables in $\mathbb{R}^n$		Variables in $\mathbb{Z}^n$	
D	$\{e_1, ..., e_n\}$	$\{He_1, ..., He_n\}$	$\{e_1, ..., e_n\}$	$\{He_1, ..., He_n\}$
eval	Min–max	Min–max	Min–max	Min–max
	Average	Average	Average	Average
	213–569	265–468	4686–9609	41–83
	382.6	345.4	7238.4	62.8
glob	**77%**	**100%**	**100%**	**100%**

Table 2 Performance for function (53), n = 50

var	Variables in $\mathbb{R}^n$		Variables in $\mathbb{Z}^n$	
D	$\{e_1, ..., e_n\}$	$\{He_1, ..., He_n\}$	$\{e_1, ..., e_n\}$	$\{He_1, ..., He_n\}$
eval	Min–max	Min–max	Min–max	Min–max
	Average	Average	Average	Average
	483,349–1,435,054	62,566–998,547	5345–10,597	5206–9850
	944,546.8	727,719.2	7031.9	7793.3
glob	**93%**	**86%**	**100%**	**100%**

The version shown above has an integer global solution. The results are given in Table 1.

6.2 Rosenbrock function [9]. Optimizer: $x^k = 1, k = 1, \dots, 50$

$$f(x) = \sum_{k=1}^{k=n-1} 100(x^{k+1} - (x^k)^2)^2 + (1 - x^k)^2, \quad -5 \le x^k \le 5, k = 1, \dots, n. \tag{53}$$

The function (53) has been extensively used as a benchmark. It has long narrow valleys and local minimizers for $n > 4$ [27]. The global solution is all integer $x = (1, 1, \dots, 1, \dots)$ regardless the number of variables. The results shown in Table 2 with $x \in \mathbb{Z}^{50}$ are remarkable when compared with [38]. It never gets trapped by any of the numerous local function minimizers [27].

6.3 Lukšan function [34]. Optimizer: $y = (1,\ 1)$

$$\begin{gathered} x^1 = y^1 + 0.14, \quad x^2 = y^2 - 0.1, \\ w^1 = (x^1)^2 + (x^2)^4, \\ w^2 = (x^1 - 2)^2 + (x^2 - 2)^2, \\ w^3 = 2\exp(x^2 - x^1). \\ f(w) = \max(w^1, w^2, w^3). \end{gathered} \tag{54}$$

Table 3 Performance for function (54), n = 2

var	Variables in $\mathbb{R}^n$		Variables in $\mathbb{Z}^n$	
D	$\{e_1, ..., e_n\}$	$\{He_1, ..., He_n\}$	$\{e_1, ..., e_n\}$	$\{He_1, ..., He_n\}$
eval	Min–max	Min–max	Min–max	Min–max
	Average	Average	Average	Average
	288–616	267–714	27–66	22–57
	397.6	384.2	46.1	30.8
glob	**10%**	**80%**	**64%**	**80%**

Minimax are non-smooth problems that appear often in testing algorithms.

6.4 Pintér function [40]. Optimizer: $x = y$

$$\begin{aligned} f(x) = 0.025n \sum_{k=1}^{k=n} (x^k - y^k)^2 + \sin^2(x^k - y^k) \\ + \sin^2 \left(\sum_{k=1}^{k=n} \left[(x^k - y^k) + (x^k - y^k)^2 \right] \right), \\ -5 \leq x^k \leq 5, k = 1, \ldots, n. \end{aligned} \tag{55}$$

The Pintér problem was a difficult test for our algorithms. The author in [40, Section 4.4] justifies the need of using global search strategies. Problem (55) possesses many local minimizers with close function values. The dimension and solution can be arbitrarily selected. We chose $y^k = 1, k = 1, \ldots, n$ with $n = 5$ and $n = 10$. The results are shown on Table 4.

6.5 Ackley function [9]. Optimizer: $x^k = 0, k = 1, \ldots, 30$

$$\begin{aligned} f(x) = 20 + \exp(1) - 20 \exp \left(-0.2 \sqrt{\frac{1}{n} \sum_{k=1}^{k=n} (x^k)^2} \right) - \exp \left(\frac{1}{n} \sum_{k=1}^{k=n} \cos(2\pi x^k) \right), \\ -10 \leq x^k \leq 10, k = 1, \ldots, n. \end{aligned} \tag{56}$$

The authors in [29] could not solve (56) with any of the heuristic algorithms they tried. In [15] the author reports that his algorithm returns the global solution in 70 out of 100 runs, with 20,000 function evaluations in average. Our algorithms always found the global solution. The number of evaluations for the discrete version was remarkably low in comparison with [38].

Table 4 Performance for function (55), n = 5

var	Variables in $\mathbb{R}^n$		Variables in $\mathbb{Z}^n$	
D	$\{e_1, ..., e_n\}$	$\{He_1, ..., He_n\}$	$\{e_1, ..., e_n\}$	$\{He_1, ..., He_n\}$
eval	Min–max	Min–max	Min–max	Min–max
	Average	Average	Average	Average
	467–1889	431–1456	52–111	41–120
	841.2	560	69.1	70.3
glob	**15%**	**6%**	**4%**	**7%**

Performance for function (55), n=10

var	Variables in $\mathbb{R}^n$		Variables in $\mathbb{Z}^n$	
D	$\{e_1, ..., e_n\}$	$\{He_1, ..., He_n\}$	$\{e_1, ..., e_n\}$	$\{He_1, ..., He_n\}$
eval	Min–max	Min–max	Min–max	Min–max
	Average	Average	Average	Average
	1041–1773	873–2088	135–153	212–232
	1304.2	1165.3	151.6	222
glob	**1%**	**5%**	**1%**	**2%**

Table 5 Performance for function (56), n = 30

var	Variables in $\mathbb{R}^n$		Variables in $\mathbb{Z}^n$	
D	$\{e_1, ..., e_n\}$	$\{He_1, ..., He_n\}$	$\{e_1, ..., e_n\}$	$\{He_1, ..., He_n\}$
eval	Min–max	Min–max	Min–max	Min–max
	Average	Average	Average	Average
	125,230–157,430	11,744–148,939	964–1468	1210–2157
	143,188	128,545.5	1263.2	1459.1
glob	**100%**	**100%**	**100%**	**100%**

6.6 Shekel function [9]. Optimizer: $x^k = 4, k = 1, \ldots, 4$

$$b = (0.1, 0.2, 0.2, 0.4, 0.4, 0.6, 0.3, 0.7, 0.5, 0.5),$$

$$C = \begin{bmatrix} 4, 1, 8, 6, 3, 2, 5, 8, 6, 7 \\ 4, 1, 8, 6, 7, 9, 3, 1, 2, 3 \\ 4, 1, 8, 6, 3, 2, 5, 8, 6, 7 \\ 4, 1, 8, 6, 7, 9, 3, 1, 2, 3 \end{bmatrix}.$$

$$f(x) = -\sum_{j=1}^{j=10} \left(\sum_{k=1}^{k=4} (x^k - C^{kj})^2 + b^j \right)^{-1}, \tag{57}$$

$$0 \le x^k \le 10, k = 1, \ldots, 4.$$

The Shekel function (57) has 10 local discrete minimizers, which is a challenge for the GREEDY() algorithm. Again, in regards to function evaluations, the discrete version outperformed both the continuous version and the results reported in [15, 38].

Table 6 Performance for function (57), n = 4

var	Variables in $\mathbb{R}^n$		Variables in $\mathbb{Z}^n$	
D	$\{e_1, ..., e_n\}$	$\{He_1, ..., He_n\}$	$\{e_1, ..., e_n\}$	$\{He_1, ..., He_n\}$
eval	Min–max	Min–max	Min–max	Min–max
	Average	Average	Average	Average
	400–1102	398–1291	47–61	40–109
	484.2	614.6	51.9	69.6
glob	**0%**	**21%**	**10%**	**15%**

Table 7 Performance for function (58), n = 25

var	Variables in $\mathbb{R}^n$		Variables in $\mathbb{Z}^n$	
D	$\{e_1, ..., e_n\}$	$\{He_1, ..., He_n\}$	$\{e_1, ..., e_n\}$	$\{He_1, ..., He_n\}$
eval	Min–max	Min–max	Min–max	Min–max
	Average	Average	Average	Average
	50,121–1,133,314	25,527–439,413	4202–7025	3731–7217
	650563.3	51395.1	5694.3	5757.1
glob	**60%**	**92%**	**36%**	**46%**

6.7 NgZhang function [38]. Optimizer: $x^k = 1, k = 1, \dots, 25$

$$\begin{aligned} f(x) = (x^1-1)^2 + (x^2-1)^2 + n\textstyle\sum_{k=1}^{n-1}(n-k)(x^{k+1}-(x^k)^2), \\ -5 \le x^k \le 5,\ k = 1, \dots, n. \end{aligned} \tag{58}$$

6.8 Beam function [10]. Known solution: $x = (7, 0.1, 9.4848, 0.1)$, $f(x) = 97.7$

$$\begin{aligned} &a = 36, b = 1000, c = 10^7, \\ &h = x^2(x^1-2x^4)^3/12 + 2\left[x^3(x^4)^3 + x^4x^3(x^1-x^4)^2/4\right], \\ &f(x^1, x^2, x^3, x^4) = a\left[2x^4x^3 + (x^1-2x^4)x^2\right]. \\ &\underset{x\in G}{\text{minimize}} \quad f(x) : \mathbb{R}^4 \to \mathbb{R} \\ &G = \{x \in \mathbb{R}^4 : g^1(x) = abx^1/(2h) - 5000 \le 0, \\ &\qquad g^2(x) = 36^3 b/(3ch) - 0.1 \le 0, \\ &\qquad 3.0 \le x^1 \le 7.0, \\ &\qquad 0.1 \le x^2 \le 2.0, \\ &\qquad 2.0 \le x^3 \le 12.0, \\ &\qquad 0.1 \le x^4 \le 1.0\}. \end{aligned} \tag{59}$$

The problem (59) is a real problem. The goal is to design a minimum volume solution of a beam satisfying physical constraints. A detailed description is given in [10]. To solve (59) we use the exact penalty function

$$500\big[(\max(g^1(x),0)+\max(g^2(x),0)\big].$$

The algorithm's rate of success was 72% with the search directions given by Algorithm 2. Besides, the algorithm unveiled a better solution:

$x = (6.9999198,\ 0.1000450,\ 9.4756010,\ 0.1000151)$,
$f(x) = 92.72525,\ g^1(x) = -0.00351,\ g^2(x) = -0.09383$

6.9 Mixed beam

We now mention the mixed variable problem suggested in [10]; x^4 becomes discrete with the additional constraint

$$x^4 \in V = \{0.1, 0.25, 0.35, 0.5, 0.65, 0.75, 0.9, 1.0\}. \tag{60}$$

To handle (60) as a mixed variable problem we employ an integer variable z whose values are linked to x^4 as follows:

Problem (59) is now a non-linearly constrained problem with 3 continuous variables (x^1, x^2, x^3) and the integer variable z. This kind of problem was not an aim of this paper. Nonetheless, MAIN() solved it with less than 600 function evaluations.

6.10 Comments on the numerical results

Problems 6.1–6.7 were taken from the open literature. They were originally proposed in benchmarks for continuous optimization. Nowadays many researches have added the constraint $x \in \mathbb{Z}^n$ for testing integer optimization algorithms. With this sample we hope to identify features to be improved.

Problem 6.1 The version of the Branin function exposed here has one global minimizer and two more local minimizers. This is the only problem in the

Table 8 Performance for function (59), n = 4

var	Continuous variables		Discrete variables	
D	$\{e_1, ..., e_n\}$	$\{He_1, ..., He_n\}$	$\{e_1, ..., e_n\}$	$\{He_1, ..., He_n\}$
eval	Min–max	Min–max		
	Average	Average		
	279–582	421–2378	NOT APPLICABLE	
	326.6	675.1		
glob	**0%**	**72%**		

$z =$	1	2	3	4	5	6	7	8
$x^4 =$	0.1	0.25	0.35	0.5	0.65	0.75	0.9	1.0

sample that solved the discrete version in fewer function evaluations than those needed to solve the continuous version.

Problem 6.2 The authors in [38] report that they successfully solved the discrete version of the Rosenbrock function in ≈1.6 million function evaluations. Our result is remarkable.

Problem 6.3 For the Lukšan function the results in $\mathbb{R}^n$ and coordinate search were not appealing. This is probably linked to the geometry of the problem.

Problem 6.4 The Pintér problem was the most difficult to solve. Often, the algorithms converge to a local minimizer. Global strategies are needed to improve the algorithm's performance.

Problem 6.5 In 400 runs our algorithms always returned the global minimizer of the Ackley function, This is an impressive result because few function evaluations were needed for solving the discrete version.

Problem 6.6 We had tested the Shekel function in [15]. The results have improved.

Problem 6.7 This problem was efficiently solved; however we should pay attention to the dispersion in function evaluations showed in $\mathbb{R}^n$ with random directions.

Overall, our algorithms, especially the discrete version, look promising. Better results should be expected by introducing globalization strategies.

7 Conclusion and final remarks

We have formalized and described a common framework for a class of nonmonotone Direct Search Methods. Convergence of non-smooth functions is proved with no differentiability assumptions. A new concept of quasi descent direction is included to show *nonsmooth* convergence. These methods converge under weak assumptions to classical first-order stationary points of unconstrained problems with continuous variables. Convergence prevails for box constrained optimization with mixed variables. Derivative information is not required. For unconstrained problems with continuous variables we prove convergence to a point that does not have a quasi descent direction. The role of a finite set of n orthogonal search directions randomly generated was crucial to ensure this result. Orthogonality was also an asset for the design of a variable separation algorithm for solving mixed-integer optimization problem. All convergence properties imply convergence to a Clarke stationary point, provided the function is Lipschitz continuous.

For the sake of clarity, we have exposed the theory complemented with a basic but unspecified implementation of the algorithms. Our framework is open to many variants that deserve further research. Nonetheless, we wrote a preliminary code and carried out numerical experiments on several academic problems and a 5-variable real application.

The results are encouraging. The next apparent line of research is the convergence analysis of DSMs to a nonlinearly constrained mixed-integer optimization problem.

We have left out for future study some important issues that are beyond the aims of this paper, like strategies for accelerating convergence, global minimization, parallelism, and hybrid methods. A clear research topic is the analysis and impact of non-monotone features in other successful approaches to DFO, like MADS [7], surrogate functions [25, 26, 41], and others.

In this paper, and also in [15], feasibility of the sequence of the solution estimates was forced at all iterations. If this condition can be removed, we would probably broaden the class of nonmonotone DSMs converging under the same conditions stated herein.

Acknowledgements Remark 1 and Corollary 3 were suggested by my colleagues at the *Universidade de Vigo*, Dr. José Ramón Fernández and Dr. Ramón González. Dr. Fernández advise was crucial for proving Lemma 7. Dr. Ildemaro García, from *Universidad Simón Bolívar*, helped to develop the pseudocodes. Warmest thanks to the Journal reviewers, who were enthusiastic in their reading and gave advice that enhanced the quality of this paper. The data generated and analysed during the current study constitute an excerpt of research in progress but are available from the author.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature. The Universidade de Vigo and the Consorcio Interuniversitario de Galicia (CISUG) financed this research. Both institutions covered the open access charge.

References

1. Abramson, M.A., Audet, C., Chrissis, J.W., Walston, J.G.: Mesh adaptive direct search algorithms for mixed variable optimization. Optim. Lett. **3**, 35–47 (2009)
2. Abramson, M.A., Audet, C., Le Digabel, S.: Orthomads: a deterministic MADS instance with orthogonal directions. SIAM J. Optim. **22**(2), 948–966 (2009)
3. Audet, C., Dennis Jr, J.E., Le Digabel, S.: Parallel space decomposition of the mesh adaptive direct search algorithm. SIAM J. Optim. **19**(3), 1150–1170 (2008)
4. Audet, C., Ianni, A., Le Digabel, S., Tribes, C.: Reducing the number of function evaluations in mesh adaptive direct search algorithms. SIAM J. Optim. **24**(2), 621–642 (2014)
5. Audet, C., Ianni, A., Le Digabel, S., Tribes, C.: Robust optimization of noisy blackbox problems using the mesh adaptive direct search algorithm. Optim. Lett. **12**(4), 675–689 (2018)
6. Audet, C., Dennis Jr, J.E.: Mesh adaptive direct search algorithms for constrained optimization. SIAM J. Optim. **17**(1), 188–217 (2006)
7. Audet, C., Le Digabel, S., Tribes, C.: The mesh adaptive direct search algorithm for granular and discrete variables. SIAM J. Optim. **29**(2), 1164–1189 (2018)
8. Belotti, P., Kirches, C., Leyffer, S.: Mixed-integer nonlinear optimization. Acta Numer. **22**, 1–131 (2013)
9. Bingham, D.: Virtual library of simulation experiments: test functions and datasets. sfu.ca/ ssurjano/index.html (2017)

10. Chase, N., Redemacher, M., Goodman, E., Averill, R., Sidhu, R.: A Benchmark Study of Optimization Search Algorithms, pp. 1–15. Red Cedar Technology, East Lansing (2010)
11. Dai, Y.-H.: On the nonmonotone line search. J. Optim. Theory Appl. **112**(2), 315–330 (2002)
12. Fasano, G., Liuzzi, G., Lucidi, S., Rinaldi, F.: A linesearch-based derivative-free approach for nonsmooth constrained optimization. SIAM J. Optim. **24**(3), 959–992 (2014)
13. Friedlander, A.: Elementos de programacão não-linear (1994). https://www.ime.unicamp.br/friedlan/~livro.htm
14. Garcia-Palomares, U.M.: Non monotone algorithms for unconstrained minimization: upper bounds on function values. In: Ceragioli, F. (ed.) Proceedings of the 22nd IFIP TC7 Conference, Torino, Italy, pp. 91–100. Springer. ISBN 0-387-32774-6 (2006)
15. García-Palomares, U.M.: Non-monotone derivative-free algorithm for solving optimization models with linear constraints: extensions for solving nonlinearly constrained models via exact penalty methods. TOP (2020). https://doi.org/10.1007/s11750-020-00549-y
16. García-Palomares, U.M., García-Urrea, I.J., Rodríguez-Hernández, P.S.: On sequential and parallel non-monotone derivative-free algorithms for box constrained optimization. Optim. Methods Softw. **28**(6), 1233–1261 (2013)
17. García-Palomares, U.M., González-Castaño, F.J., Burguillo-Rial, J.C.: A combined global and local search (CGLS) approach to global optimization. J. Glob. Optim. **34**(3), 409–426 (2006)
18. García-Palomares, U.M., Rodríguez, J.F.: New sequential and parallel derivative-free algorithms for unconstrained minimization. SIAM J. Optim. **13**(1), 79–96 (2002)
19. García-Palomares, U.M., Rodríguez-Hernández, P.S.: Unified approach for solving box-constrained models with continuous or discrete variables by non monotone direct search methods. Optim. Lett. **13**(1), 95–111 (2019)
20. González-Castaño, F.J., Costa-Montenegro, E., Burguillo-Rial, J.C., García-Palomares, U.M.: Outdoor wlan planning via non-monotone derivative-free optimization: algorithm adaptation and case study. Comput. Optim. Appl. **40**(3), 405–419 (2008)
21. Gould, N., Orban, D., Toint, P.: GALAHAD a library of thread-safe Fortran 90 packages for large scale nonlinear optimization. Trans. ACM Math. Softw. **29**(4), 353–372 (2003)
22. Gratton, S., Royer, C.W., Vicente, L.N., Zhang, Z.: Direct search based on probabilistic feasible descent for bound and linearly constrained problems. Comput. Optim. Appl. **72**(3), 525–559 (2019)
23. Grippo, L., Lampariello, F., Lucidi, S.: A nonmonotone line search technique for Newton's method. SIAM J. Numer. Anal. **23**(4), 707–716 (1986)
24. Gu, N., Mo, J.: Incorporating nonmonotone strategies into the trust region method for unconstrained optimization. Comput. Math. Appl. **55**(9), 2158–2172 (2008)
25. Gumma, E., Ali, M.M., Hashim, M.: A derivative-free algorithm for non-linear optimization with linear equalities constraints. Optimization **69**, 1361–1387 (2019)
26. Gumma, E., Hashim, M., Ali, M.M.: A derivative-free algorithm for linearly constrained optimization problems. Comput. Optim. Appl. **57**(3), 599–621 (2014)
27. Kok, S., Sandrock, C.: Locating and characterizing the stationary points of the extended Rosenbrock function. Evol. Comput. **17**(3), 437–453 (2009)
28. Kronqvist, J., Bernal, D.E., Lundell, A., Grossmann, I.E.: A review and comparison of solvers for convex MINLP. Optim. Eng. **20**, 397–455 (2019)
29. Laguna, M., Martí, R.: Experimental testing of advanced scatter search designs for global optimization of multimodal functions. Technical report, 2002
30. Lewis, R.M., Torczon, V.: A globally convergent augmented Lagrangian pattern search algorithm for optimization with general constraints and simple bounds. SIAM J. Optim. **12**(4), 1075–1089 (2002)
31. Leyffer, S.: Integrating SQP and branch-and-bound for mixed integer nonlinear programming. Comput. Optim. Appl. **18**(3), 295–309 (2001)
32. Liu, B.: Probability of pairwise difference of samples from distribution with finite support (2013). https://stats.stackexchange.com/users/31162/brandonliu
33. Liuzzi, G., Lucidi, S., Rinaldi, F.: Derivative-free methods for mixed-integer constrained optimization problems. J. Optim. Theory Appl. **164**(3), 933–965 (2015)
34. Lukšan, L., Vlček, J.: Test problems for nonsmooth unconstrained and linearly constrained optimization. Report 798, Academy of Sciences, Czech Republic (2020)
35. Naevdal, G.: Positive bases with maximal cosine measure. Optim. Lett. **13**(6), 1381–1388 (2019)
36. Newby, E., Ali, M.M.: A trust-region-based derivative free algorithm for mixed integer programming. Comput. Optim. Appl. **60**(1), 199–229 (2015)

37. Ng, C.K., Li, D., Zhang, L.S.: Discrete global descent method for discrete global optimization and nonlinear integer programming. J. Glob. Optim. **37**(3), 357–379 (2007)
38. Ng, C.K., Zhang, L.S., Li, D., Tian, W.W.: Discrete filled function method for discrete global optimization. Comput. Optim. Appl. **31**(1), 87–115 (2005)
39. Pachón, Á., García-Palomares, U.M.: Mid-term frequency domain scheduler for resource allocation in wireless mobile communications systems. Comput. Commun. **97**, 96–110 (2017)
40. Pintér, J.D.: Global optimization: software, test problems, and applications. In: Pardalos, P.M., Romeijn, E.H. (eds.) Handbook of Global Optimization, vol. 2, Chap. 15, pp. 515–569. Springer, New York (2013)
41. Powell, M.J.D.: The BOBYQA Algorithm for Bound Constrained Optimization Without Derivatives. Technical report DAMTP2009/NA06, Department of Applied Mathematics and Theoretical Physics, University of Cambridge (2009)
42. Torczon, V.: On the convergence of pattern search algorithms. SIAM J. Optim. **7**(1), 1–25 (1997)
43. Woon, S.F., Rehbock, V.: A critical review of discrete filled function methods in solving nonlinear discrete optimization problems. Appl. Math. Comput. **217**, 25–41 (2010)
44. Zhang, H., Hager, W.W.: A nonmonotone line search technique and its application to unconstrained optimization. SIAM J. Optim. **14**(4), 1043–1056 (2004)
45. Zhou, J.L., Tits, A.L.: Nonmonotone line search for minimax problems. J. Optim. Theory Appl **3**, 455–476 (1993)