J Optim Theory Appl (2015) 164:933–965
DOI 10.1007/s10957-014-0617-4

Derivative-Free Methods for Mixed-Integer Constrained Optimization Problems

Giampaolo Liuzzi · Stefano Lucidi · Francesco Rinaldi

Received: 3 September 2013 / Accepted: 21 June 2014 / Published online: 15 July 2014

Abstract Methods which do not use any derivative information are becoming popular among researchers, since they allow to solve many real-world engineering problems. Such problems are frequently characterized by the presence of discrete variables, which can further complicate the optimization process. In this paper, we propose derivative-free algorithms for solving continuously differentiable Mixed Integer Non-Linear Programming problems with general nonlinear constraints and explicit handling of bound constraints on the problem variables. We use an exterior penalty approach to handle the general nonlinear constraints and a local search approach to take into account the presence of discrete variables. We show that the proposed algorithms globally converge to points satisfying different necessary optimality conditions. We report a computational experience and a comparison with a well-known derivative-free optimization software package, i.e., NOMAD, on a set of test problems. Furthermore, we employ the proposed methods and NOMAD to solve a real problem concerning the optimal design of an industrial electric motor. This allows to show that the method converging to the better extended stationary points obtains the best solution also from an applicative point of view.

G. Liuzzi (✉)
Istituto di Analisi dei Sistemi ed Informatica (IASI) "A.Ruberti", CNR, Viale Manzoni 30, 00185 Rome, Italy
e-mail: giampaolo.liuzzi@iasi.cnr.it

S. Lucidi
Dipartimento di Informatica e Sistemistica "A. Ruberti", "Sapienza" Università di Roma, Via Ariosto 25, 00185 Rome, Italy
e-mail: lucidi@dis.uniroma1.it

F. Rinaldi
Dipartimento di Matematica, Università di Padova, Via Trieste 63, 35121 Padua, Italy
e-mail: rinaldi@math.unipd.it

Keywords Mixed integer nonlinear programming · Derivative-free optimization · Nonlinear constrained optimization

Mathematics Subject Classification 90C11 · 90C30 · 90C56

1 Introduction

Many engineering applications that can be modeled as Mixed Integer Nonlinear Programming (MINLP) problems have a twofold difficulty. On the one hand, the objective and nonlinear constraint functions are of the black-box type, so that first-order derivatives are not available (see [1] for a recent survey on derivative-free methods). On the other hand, the presence of discrete variables requires an ad-hoc treatment. To the best of our knowledge, there exist only a few papers describing derivative-free algorithms for MINLP problems. In [2], mesh adaptive direct search (MADS) algorithms, originally proposed in [3], have been extended to handle categorical variables. The extension to mixed variable programming for generalized pattern search (GPS) algorithms, described and analyzed in [4–6], has been proposed in [7,8] for bound-constrained problems. Successively, in [9,10], the filter GPS approach for nonlinear constrained problems [11] has been extended to discrete variables. Furthermore, we cite the paper [12], where a definition of implicitly and densely discrete problems is considered, namely, problems where the variables lie implicitly in an unknown "discrete" closed set (i.e., a closed set of isolated points in $\mathbb{R}^n$). In [12], a modification of a direct search algorithm is presented to tackle this kind of problems, and a theoretical analysis is reported. In [13], a linesearch strategy for linearly constrained problems [14] is adopted for the solution of Problem mixed variable problems. In [15], the derivative-free algorithms proposed in [16] are extended to the solution of mixed variable problems with bound constraints only. In [17], a probabilistic method using surrogate models for the optimization of computationally expensive mixed integer black-box problems is proposed. The method is proved to be convergent to the global optimum with probability one. Finally, in the recent paper [18], a scatter search procedure is proposed to solve black-box optimization problems, where all of the variables can only assume integer values.

In this paper, we extend the approach proposed in [19] for box-constrained, mixed integer problems using a sequential quadratic penalty approach described and analyzed in [20]. The presence of both integer variables and nonlinear constraints makes the extension of the approaches proposed in [19] not straightforward. In particular, the possible alternation of minimizations of continuous and discrete variables needs new theoretical analysis of the algorithms. In our framework, continuous variables are managed by means of a linesearch strategy with sufficient decrease acceptability criterion (see, e.g., [21]). Discrete variables are tackled by suitably developed Local Search procedures, which basically hinge on exploring adaptively determined discrete neighborhoods of points.

We note that the use of a linesearch procedure needs the satisfaction of a sufficient decrease in the new point generated along the search direction, which is a stronger requirement with respect to the simple decrease accepted by, e.g., pattern search meth-

ods [4–6]. However, it should be noted that the stronger requirement imposed by the sufficient decrease condition (over the simple one) does not necessarily bring to the definition of less efficient algorithms both in terms of number of function evaluations and of final function value attained by the search routine. Furthermore, as recently evidenced in [22], the imposition of a sufficient decrease condition, like the one adopted in the present paper, allows to derive a worst-case complexity bound on the number of iteration of a direct search algorithm. This worst-case complexity bound, which is the number of iterations needed to drive the norm of the objective gradient below a prefixed accuracy , is similar to the one obtained for the steepest descent method in [23] in the presence of first-order derivatives. On the contrary, if a simple decrease condition is imposed, then the worst-case complexity bound on the number of iterations seems only provable under additional strong conditions, like the objective function satisfying an appropriate decrease rate.

The paper is organized as follows. In Sect. 2, some definitions and relevant notations are introduced. Sections 3 and 4 are the main part of the paper and are devoted to the definition and analysis of two different algorithms for the solution of a MINLP problem. A computational experience of the methods proposed and comparison with NOMAD is reported in Sect. 5 both on analytic test problems and on a real optimal design problem. Finally, in Sect. 6, we draw some conclusions and discuss future developments.

2 Definitions and Notations

In the paper, we consider the following MINLP problem

$$\begin{aligned} \min\ & f(x) \\ & g(x) \leq 0 \\ & l \leq x \leq u \\ & x_i \in \mathbb{Z}, \quad i \in I_z, \end{aligned} \tag{1}$$

where $x, l, u \in \mathbb{R}^n$, $I_z \subset \{1, \ldots, n\}$, $f : \mathbb{R}^n \to \mathbb{R}$ and $g_j : \mathbb{R}^n \to \mathbb{R}$, $j = 1, \ldots, m$. Furthermore, we define $I_c := \{1, \ldots, n\} \setminus I_z$.

We assume the objective and general nonlinear constraint functions to be continuously differentiable with respect to x_i, $i \notin I_z$, even though first-order derivatives will not be used. To simplify the mathematical analysis of the proposed methods, we require $-\infty < l_i < u_i < +\infty$, for all $i = 1, \ldots, n$. Then, let us introduce

$$\mathcal{X} := \{x \in \mathbb{R}^n : l \leq x \leq u\}, \quad \mathcal{F} := \{x \in \mathbb{R}^n : g(x) \leq 0\} \cap \mathcal{X},$$
$$\mathcal{Z} := \{x \in \mathbb{R}^n : x_i \in \mathbb{Z}, i \in I_z\}.$$

For any vector $v \in \mathbb{R}^n$, we denote by $v_c \in \mathbb{R}^{|I_c|}$ and $v_z \in \mathbb{R}^{|I_z|}$ the subvectors

$$v_c := [v_i]_{i \in I_c}, \qquad v_z := [v_i]_{i \in I_z}.$$

Furthermore, for every continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$, we use the notation $\nabla_c h(x) \in \mathbb{R}^{|I_c|}$ to denote the gradient of the function with respect to the continuous variables, namely

$$\nabla_c h(x) := \left[\frac{\partial h(x)}{\partial x_i}\right]_{i \in I_c}.$$

Now we give different definitions of neighborhoods that correspond to variations of continuous and discrete variables. These are necessary since the characterization of local minimum points in mixed problems strongly depends on the particular neighborhood we use.

Hence, we introduce, for any point $\bar{x} \in \mathbb{R}^n$ and $\rho > 0$, the following:

$$\begin{aligned} \mathcal{B}_c(\bar{x}, \rho) &:= \left\{x \in \mathbb{R}^n : x_z = \bar{x}_z,\ \|x_c - \bar{x}_c\| \leq \rho\right\}, \\ \mathcal{B}_z(\bar{x}) &:= \{x \in \mathcal{Z} : x_c = \bar{x}_c,\ \|x_z - \bar{x}_z\| \leq 1\}. \end{aligned}$$

Now we can define a local minimum point for Problem (1).

Definition 2.1 (*Local minimum point*) A point $x^\star \in \mathcal{F} \cap \mathcal{Z}$ is a local minimum of Problem (1) iff, for some $\epsilon > 0$,

$$f(x^\star) \leq f(x), \quad \forall x \in \mathcal{B}_c(x^\star; \epsilon) \cap \mathcal{F},$$

$$f(x^\star) \leq f(x), \quad \forall x \in \mathcal{B}_z(x^\star) \cap \mathcal{F}.$$

It is possible to give a different definition of local minimum, which has stronger property with respect to discrete variables.

Definition 2.2 (*Extended local minimum point*) A point $x^\star \in \mathcal{F} \cap \mathcal{Z}$ is an extended local minimum of Problem (1) iff

(i) $x^\star$ is a local minimum;
(ii) every point $\bar{x} \in \mathcal{B}_z(x^\star) \cap \mathcal{F}$, with $\bar{x} \neq x^\star$, such that $f(\bar{x}) = f(x^\star)$ is a local minimum as well.

Now, we introduce the following sets of directions that will be used to describe the main theoretical results related to the MINLP algorithms proposed in the paper.

$$D := \{\pm e_1, \ldots, \pm e_n\}, \quad D_c := \{\pm e_i : i \in I_c\}, \quad D_z := \{\pm e_i : i \in I_z\},$$

where $e_i, i = 1, \ldots, n$, is the unit coordinate vector. Given $x \in \mathcal{X}$, we denote by

$$L(x) := \{i \in \{1, \ldots, n\} : x_i = l_i\}, \qquad U(x) := \{i \in \{1, \ldots, n\} : x_i = u_i\}.$$

Given $x \in \mathcal{X}$, let

$$D(x) := \{d \in \mathbb{R}^n : d_i \geq 0,\ \forall i \in L(x),\ d_i \leq 0,\ \forall\, i \in U(x)\}.$$

The following two technical propositions are reported from [20] and [24], to which we refer the interested reader for their proofs.

Proposition 2.1 *For every $x \in \mathcal{X}$, it results*

$$cone\{D \cap D(x)\} = D(x).$$

Proposition 2.2 *Let $\{x_k\}$ be a sequence of points such that $x_k \in \mathcal{X}$ for all k, and $x_k \to \bar{x}$ for $k \to \infty$. Then, for k sufficiently large,*

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

Throughout the paper, we consider the following assumptions to hold true.

Assumption 2.1 For every $x \in \mathcal{X}$, there exists a vector $\hat{d} \in D(x)$ such that $\hat{d}_i = 0$, for all $i \in I_z$, and

$$\nabla_c g_\ell(x)^T \hat{d}_c < 0, \quad \forall\, \ell \in I^+(x),$$

where $I^+(x) := \{i : g_i(x) \geq 0\}$.

Assumption 2.2 One of the following conditions holds

(i) for all $j = 1, \ldots, m$, we have $g_j(x) = \tilde{g}_j(x_c)$ with $\tilde{g}_j : \mathbb{R}^{|I_c|} \to \mathbb{R}$;
(ii) For every sequence of points $\{w_k\}$, with $w_k \in \mathcal{X} \cap \mathcal{Z}$ for all k, converging to the point $\bar{w} \in \mathcal{F} \cap \mathcal{Z}$, and for all the sequences $\tilde{w}_k \in \mathcal{X} \cap \mathcal{Z}$ such that, for all k

$$(w_k)_c = (\tilde{w}_k)_c, \qquad \|(w_k - \tilde{w}_k)_z\| = 1,$$

there exists an index $\tilde{k}$ such that either $\tilde{w}_k \in \mathcal{F}$ for all $k \geq \tilde{k}$ or $\tilde{w}_k \notin \mathcal{F}$ for all $k \geq \tilde{k}$.

Assumption 2.1 is quite standard and it is needed to guarantee existence and boundedness of the lagrange multipliers. We note that Assumption 2.1 is well-posed thanks to the standing assumption that $I_c \neq \emptyset$. Finally, Assumption 2.2 is more technical and specific to MINLP problems. Part (i) states that the constraints do not depend on the discrete variables. Part (ii) basically states a regularity property of sequences obtained considering points belonging to the discrete neighborhood of w_k (where w_k are the points of the sequence considered in Assumption 2.2) and is needed to force feasibility of points in the discrete neighborhood of the limit point.

In order to give stationarity conditions for Problem (1), we need to introduce the Lagrangian function associated with it, that is

$$L(x, \lambda) := f(x) + \sum_{i=1}^{m} \lambda_i g_i(x).$$

Repeating the proof of results reported in [25], we can prove the following necessary optimality conditions.

Proposition 2.3 *Let $x^\star \in \mathcal{F} \cap \mathcal{Z}$ be a local minimum of Problem (1). Then there exists a vector $\lambda^\star \in \mathbb{R}^m$ such that*

$$\nabla_c L(x^\star, \lambda^\star)^T (x - x^\star)_c \geq 0, \qquad \forall\, x \in \mathcal{X}, \tag{2}$$

$$f(x^\star) \leq f(x), \qquad \forall\, x \in \mathcal{B}_z(x^\star) \cap \mathcal{F}, \tag{3}$$

$$(\lambda^\star)^T g(x^\star) = 0, \quad \lambda^\star \geq 0. \tag{4}$$

Proposition 2.4 *Let $x^\star \in \mathcal{F} \cap \mathcal{Z}$ be an extended local minimum of Problem (1). Then there exists a vector $\lambda^\star \in \mathbb{R}^m$ such that (2), (4), and (3) are satisfied.*
Furthermore, for every point $\bar{x} \in \mathcal{B}_z(x^\star) \cap \mathcal{F}$ with $f(\bar{x}) = f(x^\star)$, there exists a vector $\bar{\lambda} \in \mathbb{R}^m$ such that the pair $(\bar{x}, \bar{\lambda})$ satisfies (2), (4), and (3).

According to Proposition 2.4, an extended minimum point of Problem (1) has to satisfy the following:

(i) it has to be stationary with respect to the continuous variables,
(ii) it must be a local minimum with respect to the discrete variables within the discrete neighborhood $\mathcal{B}_z(x^\star)$,
(iii) all the points $\bar{x} \in \mathcal{B}_z(x^\star) \cap \mathcal{F}$ such that $f(\bar{x}) = f(x^\star)$ have to satisfy requirements (i) and (ii).

Next, we define stationary points and extended stationary points for Problem (1).

Definition 2.3 (*Stationary point*) A point $x^\star \in \mathcal{F} \cap \mathcal{Z}$ is a stationary point of Problem (1) iff a vector $\lambda^\star \in \mathbb{R}^m$ exists such that the pair $(x^\star, \lambda^\star)$ satisfies (2), (4), and (3).

Definition 2.4 (*Extended stationary point*) A point $x^\star \in \mathcal{F} \cap \mathcal{Z}$ is an extended stationary point of Problem (1) iff a vector $\lambda^\star \in \mathbb{R}^m$ exists such that the pair $(x^\star, \lambda^\star)$ satisfies (2), (4), and (3), and for all $\bar{x} \in \mathcal{B}_z(x^\star) \cap \mathcal{F}$ such that $f(\bar{x}) = f(x^\star)$, it is possible to find a $\bar{\lambda} \in \mathbb{R}^m$ so that

$$\nabla_c L(\bar{x}, \bar{\lambda})^T (x - \bar{x})_c \geq 0, \qquad \forall\, x \in \mathcal{X}, \tag{5}$$

$$f(\bar{x}) \leq f(x), \qquad \forall\, x \in \mathcal{B}_z(\bar{x}) \cap \mathcal{F}, \tag{6}$$

$$(\bar{\lambda})^T g(\bar{x}) = 0, \quad \bar{\lambda} \geq 0. \tag{7}$$

In the paper, we consider the following penalty function:

$$P(x; \epsilon) := f(x) + \frac{1}{\epsilon} \sum_{i=1}^{m} \max\{0, g_i(x)\}^q,$$

where $q > 1$. We also introduce the following approximation of multiplier functions.

$$\lambda_j(x; \epsilon) := \frac{q}{\epsilon} \max\{0, g_j(x)\}^{q-1}, \qquad \forall\, j = 1, \ldots, m. \tag{8}$$

We are now ready to define different algorithms for the solution of Problem (1) and to analyze their convergence properties. The first algorithm (i.e., DFL) is convergent

toward stationary points of the problem. It explores the coordinate directions and updates the iterate whenever a sufficient reduction of the penalty function is found. Hence it performs a minimization of the penalty function distributed along all the variables. The second algorithm (EDFL), which is convergent to extended stationary points, is based on a Local Search procedure that is devised to better investigate the discrete neighborhoods.

3 A Linesearch Algorithm Model

In this section, we define a first Derivative-Free Linesearch algorithm (DFL) for MINLP problems. The proposed method combines two basic ingredients, which are a derivative-free optimization for bound-constrained, mixed integer problems and a penalty function approach for handling of nonlinear constraints. In particular, integer variables are tackled by a Discrete search procedure which is similar to the one defined in [19]. The presence of nonlinear constraints is accounted for by means of a derivative-free sequential penalty approach like that described and analyzed in [20].

3.1 Algorithm DFL

The main parts of the method are the *Continuous search* and *Discrete search* procedures. The Continuous search and Discrete search procedures, which investigate the corresponding coordinate direction, are similar to those described in [19], but they are applied to the penalty function $P(x;\epsilon)$. At the end of every main iteration, the algorithm computes the new values both for the penalty parameter and the sufficient decrease parameter, which are fundamental ingredients in the MINLP case as they allow us to guarantee the convergence of the proposed algorithm. The idea is that, when no discrete variable has been updated during the iteration and the tentative steps for discrete variables are all equal to one, the method updates the sufficient decrease parameter and then checks if the penalty parameter has to be updated.

The scheme of the proposed algorithm is reported in Algorithm 3.1. The *Continuous search* and *Discrete search* (see [19]) are reported in Procedures 3.2 and 3.3, respectively.

As it can be seen, Algorithm DFL performs derivative-free searches along the coordinate directions by means of two different procedures that depend on the current coordinate type, namely the *Continuous Search* and *Discrete Search* procedures. When the coordinate is continuous, that is, $i \in I_c$, the stepsize α_k^i and the tentative stepsize $\tilde{\alpha}_k^i$ are computed as described in [21]. On the other hand, when the coordinate is discrete, that is, $i \in I_z$, a kind of "discrete" linesearch is carried out by the method. This discrete linesearch is characterized by a sufficient reduction, controlled by the parameter ξ_k. When all the coordinate directions have been explored (inner `for` loop, i.e., Steps

Algorithm 3.1 Derivative-Free Linesearch (DFL) Algorithm

Data. $\theta \in]0,1[$, $\epsilon_0 > 0$, $\xi_0 > 0$, η_0, $x_0 \in X \cap \mathcal{Z}$, $\tilde{\alpha}_0^i > 0$, $i \in I_c$, $\tilde{\alpha}_0^i := 1$, $i \in I_z$, and set $d_{-1}^i := e^i$, for $i = 1, \ldots, n$.

1 **For** $k = 0, 1, \ldots$

2 Set $y_k^1 := x_k$.

3 **For** $i = 1, \ldots, n$

4 **If** $i \in I_c$ **then**

5 compute α by the *Continuous Search*$(\tilde{\alpha}_k^i, y_k^i, d_{k-1}^i, \epsilon_k; \alpha, d_k^i)$

6 **If** $\alpha = 0$ **then** set $\alpha_k^i := 0$ and $\tilde{\alpha}_{k+1}^i := \theta \tilde{\alpha}_k^i$.

7 **else** set $\alpha_k^i := \alpha$ and $\tilde{\alpha}_{k+1}^i := \alpha$.

8 **else** compute α by the *Discrete Search*$(\tilde{\alpha}_k^i, y_k^i, d_{k-1}^i, \xi_k, \epsilon_k; \alpha, d_k^i)$

9 **If** $\alpha = 0$ **then** set $\alpha_k^i := 0$ and $\tilde{\alpha}_{k+1}^i := \max\{1, \lfloor \tilde{\alpha}_k^i / 2 \rfloor\}$.

10 **else** set $\alpha_k^i := \alpha$ and $\tilde{\alpha}_{k+1}^i := \alpha$.

11 Set $y_k^{i+1} = y_k^i + \alpha_k^i d_k^i$.

12 **End For**

13 **If** $(y_k^i)_z = (x_k^i)_z$ and $\tilde{\alpha}_k^i = 1$, $i \in I_z$, **then**

14 **If** $(\max_{i \in I_c}\{\alpha_k^i, \tilde{\alpha}_k^i\} \leq \epsilon_k^q)$ and $(\|g^+(x_k)\| > \eta_k)$, choose $\epsilon_{k+1} := \theta \epsilon_k$.

15 **Else** set $\epsilon_{k+1} := \epsilon_k$.

16 set $\xi_{k+1} := \theta \xi_k$,

17 **Else** set $\xi_{k+1} := \xi_k$ and $\epsilon_{k+1} := \epsilon_k$.

18 Set $\eta_{k+1} := \theta \eta_k$.

19 Find $x_{k+1} \in X \cap \mathcal{Z}$ such that $P(x_{k+1}; \epsilon_k) \leq P(y_k^{n+1}; \epsilon_k)$.

20 **End For**

3–12), the algorithm computes (by Steps 13–18) the new values ξ_{k+1}, ϵ_{k+1}, and η_{k+1} for the sufficient reduction, penalty, and feasibility violation parameters, respectively. In particular, provided that no discrete variable has been updated and that the tentative steps along discrete coordinates are equal to one, the sufficient reduction parameter is decreased. Furthermore, the procedure checks if the penalty parameter has to be updated. It is worth noting that, in Algorithm DFL, the next iterate x_{k+1} is required to satisfy the condition $f(x_{k+1}) \leq f(x_k)$. This enables to obtain x_{k+1} by minimizing suitable approximating models of the objective function, and thus possibly improving the efficiency of the overall scheme.

Procedure 3.2 Continuous search ($\tilde{\alpha}$, y, p, ϵ; α, p^+)

Data. $\gamma > 0$, $\delta \in]0, 1[$.

1 Compute the largest $\bar{\alpha}$ such that $y + \bar{\alpha}p \in X \cap \mathcal{Z}$. Set $\alpha := \min\{\bar{\alpha}, \tilde{\alpha}\}$.

2 **If** $\alpha > 0$ and $P(y + \alpha p; \epsilon) \leq P(y; \epsilon) - \gamma\alpha^2$ **then** set $p^+ := p$ and go to Step 6.

3 Compute the largest $\bar{\alpha}$ such that $y - \bar{\alpha}p \in X \cap \mathcal{Z}$. Set $\alpha := \min\{\bar{\alpha}, \tilde{\alpha}\}$.

4 **If** $\alpha > 0$ and $P(y - \alpha p; \epsilon) \leq P(y; \epsilon) - \gamma\alpha^2$ **then** set $p^+ := -p$ and go to Step 6.

5 Set $\alpha := 0$, $p^+ = p$ and **return**.

6 Let $\beta := \min\{\bar{\alpha}, (\alpha/\delta)\}$.

7 **If** $\alpha = \bar{\alpha}$ or $P(y + \beta p; \epsilon) > P(y; \epsilon) - \gamma\beta^2$ **return**.

8 Set $\alpha := \beta$ and go to Step 6.

3.2 Preliminary Convergence Results

In order to carry out the convergence analysis of Algorithm DFL, we introduce the following two sets of iteration indices:

$$K_\xi := \{k : \xi_{k+1} < \xi_k\} \subseteq \{0, 1, \ldots\}, \quad \text{and} \tag{9a}$$

$$K_\epsilon := \{k : \xi_{k+1} < \xi_k, \epsilon_{k+1} < \epsilon_k\} \subseteq K_\xi. \tag{9b}$$

Lemma 3.1 *Algorithm DFL is well defined (i.e., it produces an infinite sequence of iterates* $\{x_k\}$*).*

Proof To show that Algorithm DFL is well defined, we need to show that both the Continuous and Discrete search procedures cannot cycle between Steps 6 and 8. If this is not the case, then a sequence $\{\beta_l\}$ should exist such that

$$\lim_{l \to \infty} P(y + \beta_l p; \epsilon) = -\infty,$$

but this would contradict the assumption that set $\mathcal{X}$ is compact. □

Procedure 3.3 Discrete search ($\tilde{\alpha}$, y, p, ξ, ϵ; α, p^+)

1 Compute the largest $\bar{\alpha}$ such that $y + \bar{\alpha}p \in X \cap \mathcal{Z}$. Set $\alpha := \min\{\bar{\alpha}, \tilde{\alpha}\}$.

2 **If** $\alpha > 0$ and $P(y + \alpha p; \epsilon) \leq P(y; \epsilon) - \xi$ **then** set $p^+ := p$ and go to Step 6.

3 Compute the largest $\bar{\alpha}$ such that $y - \bar{\alpha}p \in X \cap \mathcal{Z}$. Set $\alpha := \min\{\bar{\alpha}, \tilde{\alpha}\}$.

4 **If** $\alpha > 0$ and $P(y - \alpha p; \epsilon) \leq P(y; \epsilon) - \xi$ **then** set $p^+ := -p$ and go to Step 6.

5 Set $\alpha := 0$, $p^+ = p$ and **return**.

6 Let $\beta := \min\{\bar{\alpha}, 2\alpha)\}$.

7 **If** $\alpha = \bar{\alpha}$ or $P(y + \beta p; \epsilon) > P(y; \epsilon) - \xi$ **return**.

8 Set $\alpha := \beta$ and go to Step 6.

In the following lemma, we characterize the asymptotic behavior of the sequences $\{\alpha_k^i\}$ and $\{\tilde{\alpha}_k^i\}$, $i \in \{1, \ldots, n\}$, produced by DFL.

Lemma 3.2 *Let $\{x_k\}$, $\{\xi_k\}$, $\{\epsilon_k\}$, $\{y_k^i\}$, $\{\alpha_k^i\}$, $\{\tilde{\alpha}_k^i\}$, $i \in \{1, \ldots, n\}$ be the sequences produced by Algorithm DFL. Then*

(i) If the monotonically nonincreasing sequence of positive numbers $\{\epsilon_k\}$ is such that

$$\lim_{k\to\infty} \epsilon_k = \bar{\epsilon} > 0,$$

for all $i \in I_c$, then

$$\lim_{k\to\infty} \alpha_k^i = 0,$$
$$\lim_{k\to\infty} \tilde{\alpha}_k^i = 0.$$

(ii) If the monotonically nonincreasing sequence of positive numbers $\{\epsilon_k\}$ is such that

$$\lim_{k\to\infty} \epsilon_k = 0,$$

for all $i \in I_c$, then

$$\lim_{k\to\infty, k\in K_\epsilon} \alpha_k^i = 0,$$
$$\lim_{k\to\infty, k\in K_\epsilon} \tilde{\alpha}_k^i = 0.$$

Proof For $i \in I_c$, the proof follows from Proposition 5 in [20]. □

Lemma 3.3 *Let $\{\xi_k\}$ and $\{\epsilon_k\}$ be the sequences produced by Algorithm DFL. Then*

(i)

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

(ii) the set K_ξ, defined in (9a), has infinitely many elements. Moreover, if $\lim_{k\to\infty} \epsilon_k = 0$, then also the set K_ϵ, defined in (9b), has infinitely many elements.

Proof First we prove point (i). As it can be seen, the sequence $\{\xi_k\}$ generated by Algorithm DFL is monotonically nonincreasing, that is, $0 < \xi_{k+1} \leq \xi_k$, for all k. Therefore,

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

By contradiction, we assume that $M > 0$. In this case, we would have

$$\xi_{k+1} = \xi_k = M,$$

for all $k \geq \bar{k}$, with $\bar{k} > 0$ a sufficiently large index. Then, by step 17 of Algorithm DFL, we would also have $\epsilon_{k+1} = \epsilon_k = \bar{\epsilon}$, for all $k \geq \bar{k}$. Then, by definition of Algorithm DFL and by Step 2 and 4 of the Discrete search procedure, for all $k \geq \bar{k}$, an index $\bar{\imath} \in I_z$ (depending on k) would exist such that

$$P(x_{k+1}; \bar{\epsilon}) \leq P(y_k^{\bar{\imath}} \pm \alpha_k^{\bar{\imath}} d_k^{\bar{\imath}}; \bar{\epsilon}) \leq P(y_k^{\bar{\imath}}; \bar{\epsilon}) - M \leq P(x_k; \bar{\epsilon}) - M, \tag{10}$$

otherwise the algorithm would have performed the parameter updating (i.e., $\xi_{k+1} = \theta \xi_k$). By (10), we have

$$\lim_{k \to \infty} P(x_k; \bar{\epsilon}) = -\infty,$$

and this contradicts the assumption that $P(\cdot; \bar{\epsilon})$ is continuous on the compact set $\mathcal{X}$.

Finally, we prove point (ii). Point (i) and the updating rule of parameter ξ_k in Algorithm DFL imply that the set K_ξ is infinite. Furthermore, if

$$\lim_{k \to \infty} \epsilon_k = 0,$$

the updating rule of Algorithm DFL for ξ_k and ϵ_k implies that the set K_ϵ is infinite as well. □

Lemma 3.4 *Let $\{x_k\}$ and $\{y_k^i\}$, $i = 1, \ldots, n+1$ be the sequences of points produced by Algorithm DFL and let $\tilde{K} \subseteq K_\xi$, where K_ξ is defined in (9a), be such that*

$$\lim_{k \to \infty, k \in \tilde{K}} x_k = x^*. \tag{11}$$

Then

$$\lim_{k \to \infty, k \in \tilde{K}} y_k^i = x^*, \qquad i = 1, \ldots, n+1.$$

Proof By considering limit (11), for $k \in \tilde{K}$ and sufficiently large,

$$(x_k)_z = (x^\star)_z. \tag{12}$$

Thus, we have a failure along all the search directions related to the discrete variables, and the trial steps related to those directions cannot be further reduced.

Recalling the definition of K_ξ in (9a), by the instructions of the algorithm DFL, for $k \in \tilde{K}$, we have

$$\begin{aligned} (y_k^i)_z &= (x_k)_z, \quad i = 1, \ldots, n+1 \\ \tilde{\alpha}_k^i &= 1, \quad i \in I_z. \end{aligned}$$

Recalling (12), for $k \in \tilde{K}$ and sufficiently large, we further have that

$$(y_k^i)_z = (x^\star)_z, \quad i = 1, \ldots, n+1. \tag{13}$$

Lemma 3.2 guarantees

$$\lim_{k\to\infty, k\in\tilde{K}} \alpha_k^i = 0, \quad i \in I_c, \tag{14}$$

so that by (13) and (14), we can write

$$\lim_{k\to\infty, k\in\tilde{K}} y_k^i = x^\star, \quad i = 1, \ldots, n+1,$$

which completes the proof. □

3.3 Main Convergence Results

Now we show that accumulation points exist, which are stationary in the sense of Definition 2.3. For the sake of simplicity, we first show stationarity with respect to the continuous variables and then with respect to the discrete ones.

Proposition 3.1 *Let $\{x_k\}$ be the sequence of points produced by Algorithm DFL and let K_ξ and K_ϵ defined in (9). Then,*

(i) if $\lim_{k\to\infty} \epsilon_k = \bar{\epsilon}$, then every limit point of $\{x_k\}_{K_\xi}$ is stationary for Problem (1) with respect to the continuous variables;
(ii) if $\lim_{k\to\infty} \epsilon_k = 0$, then every limit point of $\{x_k\}_{K_\epsilon}$ is stationary for Problem (1) with respect to the continuous variables.

Proof Let us consider any limit point $\bar{x}$ of the subsequence $\{x_k\}_{K_\xi}$ (point (i)) or $\{x_k\}_{K_\epsilon}$ (point (ii)). Then, for k sufficiently large

$$(x_k)_z = (\bar{x})_z. \tag{15}$$

Now, let us note that, by the instructions of Algorithm DFL, for all $k \in K_\xi$,

$$(y_k^{n+1})_z = (x_k)_z \text{ and } \tilde{\alpha}_k^i = 1, \ i \in I_z.$$

Hence, by (15), for k sufficiently large and $k \in K_\xi$ (point (i)) or $k \in K_\epsilon$ (point (ii)), the discrete variables of x_k are no longer updated.

The rest of the proof follows exactly the same reasoning as in the proof of Theorem 1 in [20]. The only difference can be found in the definition of subsequence $\{x_k\}_{\bar{K}}$. In [20],

$$\bar{K} := \{0, 1, 2, \ldots\}, \qquad \text{if } \lim_{k\to\infty} \epsilon_k = \bar{\epsilon} > 0,$$
$$\bar{K} := \{k : \epsilon_{k+1} < \epsilon_k\}, \qquad \text{if } \lim_{k\to\infty} \epsilon_k = 0.$$

Here, due to the presence of the discrete variables, we have to consider different subsequences, namely,

$$\bar{K} := K_\xi, \qquad \text{if } \lim_{k\to\infty} \epsilon_k = \bar{\epsilon} > 0,$$
$$\bar{K} := K_\epsilon, \qquad \text{if } \lim_{k\to\infty} \epsilon_k = 0,$$

where K_ξ and K_ϵ are defined as in Lemma 3.3. □

Now we prove that accumulation points exist, which are local minima with respect to the discrete variables.

Proposition 3.2 *Let $\{x_k\}$ be the sequence of points produced by Algorithm DFL. Let $K_\xi \subseteq \{1, 2, \ldots\}$ and $K_\epsilon \subseteq K_\xi$ be defined in (9). Then,*

(i) if $\lim_{k\to\infty} \epsilon_k = \bar{\epsilon}$, then every limit point $x^\star$ of $\{x_k\}_{K_\xi}$ is a local minimum for Problem (1) with respect to the discrete variables, namely $f(x^\star) \le f(\bar{x})$, for all $\bar{x} \in \mathcal{B}_z(x^\star) \cap \mathcal{F}$;

(ii) if $\lim_{k\to\infty} \epsilon_k = 0$, then every limit point $x^\star$ of $\{x_k\}_{K_\epsilon}$ is a local minimum for Problem (1) with respect to the discrete variables, namely $f(x^\star) \le f(\bar{x})$, for all $\bar{x} \in \mathcal{B}_z(x^\star) \cap \mathcal{F}$.

Proof Let us denote

$$\tilde{K} := \begin{cases} K_\xi, & \text{if } \lim_{k\to\infty} \epsilon_k = \bar{\epsilon}, \\ K_\epsilon, & \text{if } \lim_{k\to\infty} \epsilon_k = 0, \end{cases}$$

where K_ξ and K_ϵ are defined in (9). Let $x^\star$ be any accumulation point of $\{x_k\}_{\tilde{K}}$ and let $\bar{K} \subseteq \tilde{K}$ be an index set such that

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

By Lemma 3.4, we have

$$\lim_{k\to\infty, k\in\bar{K}} y_k^i = x^\star, \quad i = 1, \ldots, n+1. \tag{16}$$

Let us consider any point $\bar{x} \in \mathcal{B}_z(x^\star) \cap \mathcal{F}$. By the definition of the discrete neighborhood $\mathcal{B}_z(x)$ and of the set D_z, a direction $\bar{d} \in D(x^\star) \cap D_z$ exists such that

$$\bar{x} = x^\star + \bar{d}. \tag{17}$$

Taking into account (13) and (17), for $k \in \bar{K}$ and sufficiently large, we can write

$$(\bar{x})_z = (x^\star + \bar{d})_z = (y_k^i + \bar{d})_z, \quad i = 1, \ldots, n+1. \tag{18}$$

Now, by Proposition 2.2, we have, for $k \in \bar{K}$ and sufficiently large,

$$\bar{d} \in D(x_k) \cap D_z.$$

Therefore, there exists $d_k^{i_k}$ such that $d_k^{i_k} = \bar{d}$. As we have a finite set of search directions, we can consider, without any loss of generality, a subsequence such that $i_k = \bar{\imath}$, and we can write

$$(\bar{x})_z = (x^\star + d_k^{\bar{\imath}})_z = (y_k^{\bar{\imath}} + d_k^{\bar{\imath}})_z, \tag{19}$$

for all $k \in \bar{K}$ and sufficiently large. Thus, by (16) and (19), we can write

$$\lim_{k\to\infty, k\in\bar{K}} y_k^{\bar{\imath}} + d_k^{\bar{\imath}} = x^\star + \bar{d} = \bar{x}$$

Hence, for all $k \in \bar{K}$ and sufficiently large, by (18),

$$(y_k^{\bar{\imath}} + d_k^{\bar{\imath}})_j = (\bar{x})_j, \;\; j \in I_z. \tag{20}$$

Furthermore, for all $k \in \bar{K}$ and considering that $\bar{\imath} \in I_z$,

$$(y_k^{\bar{\imath}} + d_k^{\bar{\imath}})_j = (y_k^{\bar{\imath}})_j \;\; j \in I_c. \tag{21}$$

Then, for $k \in \bar{K}$ and sufficiently large, by (20) and (21) and recalling that $\bar{x}, \;\; y_k^{\bar{\imath}} \in \mathcal{X} \cap \mathcal{Z}$, we have

$$y_k^{\bar{\imath}} + d_k^{\bar{\imath}} \in \mathcal{X} \cap \mathcal{Z}.$$

Therefore, for $k \in \bar{K}$ and sufficiently large, the algorithm evaluates the function P in the point $y_k^{\bar{\imath}} + d_k^{\bar{\imath}}$, and obtains

$$P(y_k^{\bar{\imath}} + d_k^{\bar{\imath}}; \epsilon_k) > P(y_k^{\bar{\imath}}; \epsilon_k) - \xi_k. \tag{22}$$

Recalling the expression of the penalty function $P(x; \epsilon)$ and of the functions $\lambda_l(x; \epsilon)$ (defined in (8)), we can write

$$\begin{aligned} P(y_k^{\bar{\imath}}; \epsilon_k) &= f(y_k^{\bar{\imath}}) + \frac{1}{\epsilon_k} \sum_{l=1}^{m} \max\{0, g_l(y_k^{\bar{\imath}})\}^q \\ &= f(y_k^{\bar{\imath}}) + \frac{1}{q} \sum_{l=1}^{m} \lambda_l(y_k^{\bar{\imath}}; \epsilon_k) \max\{0, g_l(y_k^{\bar{\imath}})\}. \end{aligned}$$

Noting that points y_k^i, $i \in I_c$, satisfy the assumptions of Proposition 7.1 in Appendix and recalling that $x^\star \in \mathcal{F}$, we have (by Proposition 7.1)

$$\lim_{k\to\infty, k\in\bar{K}} \lambda_l(y_k^{\bar{\imath}}; \epsilon_k) \max\{0, g_l(y_k^{\bar{\imath}})\} = 0.$$

Therefore, we obtain

$$\lim_{k\to\infty, k\in\bar{K}} P(y_k^{\bar{l}};\epsilon_k) = f(x^\star).$$

Now, if part (i) of Assumption 2.2 holds, then we have

$$\lambda_l(y_k^{\bar{l}}+d_k^{\bar{l}};\epsilon_k)\max\{0, g_l(y_k^{\bar{l}}+d_k^{\bar{l}})\} = \lambda_l(y_k^{\bar{l}};\epsilon_k)\max\{0, g_l(y_k^{\bar{l}})\},$$

which yields

$$\lim_{k\to\infty, k\in\bar{K}} P(y_k^{\bar{l}}+d_k^{\bar{l}};\epsilon_k) = f(\bar{x}). \tag{23}$$

If, on the other hand, part (ii) of Assumption 2.2 holds, then for $k \in \bar{K}$ and sufficiently large, we have

$$\lambda_l(y_k^{\bar{l}}+d_k^{\bar{l}};\epsilon_k)\max\{0, g_l(y_k^{\bar{l}}+d_k^{\bar{l}})\} = 0$$

and, hence, we obtain again

$$\lim_{k\to\infty, k\in\bar{K}} P(y_k^{\bar{l}}+d_k^{\bar{l}};\epsilon_k) = f(\bar{x}). \tag{24}$$

Finally, by making the limit in (22) and by using (23) and (24), we obtain

$$f(\bar{x}) \geq f(x^\star)$$

which completes the proof. □

Finally, we can now derive the main theoretical result concerning the global convergence properties of Algorithm DFL.

Theorem 3.1 *Let $\{x_k\}$ and $\{\epsilon_k\}$ be the sequences generated by Algorithm DFL. Let $K_\xi \subseteq \{1, 2, \ldots\}$ and $K_\epsilon \subseteq K_\xi$ be defined in (9). Then, $\{x_k\}$ admits limit points and*

(i) if $\lim_{k\to\infty} \epsilon_k = \bar{\epsilon}$, then every limit point of $\{x_k\}_{K_\xi}$ is stationary for Problem (1);
(ii) if $\lim_{k\to\infty} \epsilon_k = 0$, then every limit point of $\{x_k\}_{K_\epsilon}$ is stationary for Problem (1).

Proof By the instructions of Algorithm DFL, every iterate x_k belongs to $\mathcal{X}$ which is compact. Hence $\{x_k\}$ admits limit points. Then, points (i) and (ii) follow by considering Propositions 3.1 and 3.2. □

4 Convergence to Extended Stationary Points

In this section, we suitably modify Algorithm DFL to ensure convergence to extended stationary points. Convergence to such points can be enforced by refining the searches along the directions related to the discrete variables. Indeed, we replace the Discrete Search used in Algorithm DFL with a new one, namely the *Local search* procedure,

which explores more deeply the discrete neighborhoods. Below we report the scheme of the Local Search procedure.

Local search$(\bar{\alpha}, y, p, \xi, \epsilon; \alpha, \tilde{z})$.

Data. $\nu > 0$.

Initialization. Compute the largest $\bar{\alpha}$ such that $y + \bar{\alpha}p \in \mathcal{X} \cap \mathcal{Z}$. Set $\alpha := \min\{\bar{\alpha}, \bar{\alpha}\}$ and $z := y + \alpha p$.

0 **If** $\alpha = 0$ or $P(z;\epsilon) > P(y;\epsilon) + \nu$ **then** Set $\tilde{z} := y$, $\alpha := 0$ and **return**.

1 **If** $\alpha > 0$ and $P(z;\epsilon) \leq P(y;\epsilon) - \xi$ **then** go to Step 2. **Else** go to Step 5.

2 Let $\beta := \min\{\bar{\alpha}, 2\alpha\}$.

3 **If** $\alpha = \bar{\alpha}$ or $P(y + \beta p;\epsilon) > P(y;\epsilon) - \xi$ **then** $\tilde{z} := y + \alpha p$ and **return**.

4 Set $\alpha := \beta$ and go to Step 2.

5 (*Grid search*) Set $z := y + \alpha p$.

6 Set $w^1 := z$.

7 **For** $i = 1, \ldots, n$

8 **If** $i \in I_z$ compute $\hat{\alpha}$ by the *Discrete Search*$(\tilde{\alpha}^i, w^i, e^i, \xi, \epsilon; \hat{\alpha}, q^i)$

9 **If** $\hat{\alpha} \neq 0$ **and** $P(w^i + \hat{\alpha}q^i;\epsilon) \leq P(y;\epsilon) - \xi$ **then**

10 set $\tilde{z} := w^i + \hat{\alpha}q^i$, $\alpha := 0$ and **return**

11 **If** $i \in I_c$ compute $\hat{\alpha}$ by the *Continuous Search*$(\tilde{\alpha}^i, w^i, e^i, \epsilon; \hat{\alpha}, q^i)$

12 **If** $\hat{\alpha} \neq 0$ **and** $P(w^i + \hat{\alpha}q^i;\epsilon) \leq P(y;\epsilon) - \xi$ **then**

13 set $\tilde{z} := w^i + \hat{\alpha}q^i$, $\alpha := 0$ and **return**

14 Set $w^{i+1} := w^i + \hat{\alpha}q^i$.

15 **End For**

16 Set $\tilde{z} := y$, $\alpha := 0$ and **return**.

In this procedure, we first verify if a point along the search direction guarantees a sufficient decrease of the penalty function. If so, we accept the point similar to Algorithm DFL. Otherwise and differently from the Discrete Search procedure, we consider two different cases:

(i) the new point is significantly worse (in terms of penalty function value) than the current one, then we discard the new point;

(ii) the new point is not significantly worse than the current one, then we perform a "grid search" (i.e., a new search both along continuous and discrete directions starting from the new point). If we find, by means of this "grid search," a new point that guarantees a sufficient decrease of the penalty function, then it becomes the current point.

Figure 1 illustrates how the Local Search procedure works in practice. We assume d_1 and d_2 to be the directions related to the discrete variables and d_3 the direction related to the continuous variable. Let us suppose that, along the discrete direction d_1, the Local Search finds a new point z that is not significantly worse than the current one y (See Fig. 1a). Then the Local Search procedure performs the Grid Search starting from z (see Fig. 1b). Finally, Fig. 1c depicts the situation in which the Grid Search

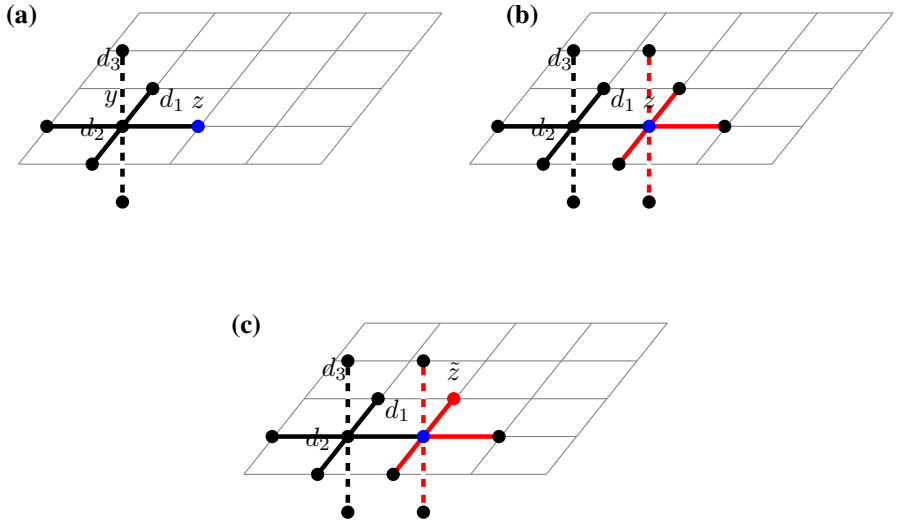

Fig. 1 How the Local Search works in practice. **a** The Local Search finds a new point z that is not significantly worse than the current one. **b** The Grid Search starts from z. **c** The Grid Search finds a point $\tilde{z}$ that guarantees a sufficient decrease of the penalty function

finds a point $\tilde{z}$ that guarantees a sufficient decrease of the penalty function with respect to the central point y.

Algorithm EDFL can be seen as an enrichment of algorithm DFL. Indeed, along the continuous directions the Continuous Search is performed, whereas the discrete directions are investigated by means of the Local Search procedure. Depending on the results of the Local Search, one case out of three is executed. They are denoted $L^1_\pm$, $L^2_\pm$, and $L^3_\pm$, where the subscript $\pm$ distinguishes if the Local Search is invoked along d^i_k or $-d^i_k$. More in particular

(i) Case L^1_+ is executed when the Local Search returns $\alpha = 0$ and $\tilde{z} \neq y^i_k$, that is, when a point yielding a sufficient decrease of the penalty function is found by the Grid Search in the Local Search procedure. In this case, the algorithm sets $\alpha^i_k = 0$, $\tilde{\alpha}^i_{k+1} = \tilde{\alpha}^i_k$, $y^{n+1}_k = \tilde{z}$, $d^i_{k+1} = d^i_k$ exit the inner **For** loop (step 3–24) and jumps directly to step 25, where ξ_{k+1}, ϵ_{k+1} and η_{k+1} are computed.

(ii) Case L^2_+ is executed when the Local Search returns $\alpha = 0$ and $\tilde{z} = y^i_k$, which means that the Local Search along direction d^i_k failed. In this case, the algorithm tries to compute α by means of the Local Search along the opposite direction $-d^i_k$.

(iii) Case L^3_+ is executed when the Local Search returns $\alpha \neq 0$, that is, when a sufficient decrease of the penalty function is achieved along direction d^i_k.

As regards cases L^1_-, L^2_-, and L^3_-, we note that L^1_- and L^3_- are similar, respectively, to L^1_+ and L^3_+. While in case L^2_-, that is, when both the Local Searches along d^i_k and $-d^i_k$ fail, the trial stepsize $\tilde{\alpha}^i_k$ is reduced, namely $\tilde{\alpha}^i_{k+1} = \max\{1, \lfloor \tilde{\alpha}^i_k/2 \rfloor\}$.

Extended Derivative-Free Linesearch (EDFL) Algorithm

Data. $\theta \in]0,1[$, $\epsilon_0 > 0$, $\xi_0 > 0$, $\eta_0 \geq 0$, $x_0 \in \mathcal{X} \cap \mathcal{Z}$, $\tilde{\alpha}_0^i > 0$, $i \in I_c$, $\tilde{\alpha}_0^i = 1$, $i \in I_z$, and set $d_{-1}^i = e^i$, for $i = 1, \ldots, n$.

1 **For** $k = 0, 1, \ldots$
2 Set $y_k^1 = x_k$.
3 **For** $i = 1, \ldots, n$
4 **If** $i \in I_c$ **then**
5 compute α by the *Continuous Search*$(\tilde{\alpha}_k^i, y_k^i, d_{k-1}^i, \epsilon_k; \alpha, d_k^i)$
6 **If** $\alpha = 0$ **then** set $\alpha_k^i = 0$ and $\tilde{\alpha}_{k+1}^i = \theta \tilde{\alpha}_k^i$.
7 **else** set $\alpha_k^i = \alpha$, $\tilde{\alpha}_{k+1}^i = \alpha$.
8 **else** compute α by the *Local Search*$(\tilde{\alpha}_k^i, y_k^i, d_{k-1}^i, \xi_k, \epsilon_k; \alpha, \tilde{z})$
9 **Case** L_+^1 ($\alpha = 0$ and $\tilde{z} \neq y_k^i$):
10 Set $\alpha_k^i = 0$, $\tilde{\alpha}_{k+1}^i = \tilde{\alpha}_k^i$, $y_k^{n+1} = \tilde{z}$, $d_k^i = d_{k-1}^i$ and **Exit For**
11 **Case** L_+^2 ($\alpha = 0$ and $\tilde{z} = y_k^i$):
12 compute α by the *Local Search*$(\tilde{\alpha}_k^i, y_k^i, -d_{k-1}^i, \xi_k, \epsilon_k; \alpha, \tilde{z})$
13 **Case** L_-^1 ($\alpha = 0$ and $\tilde{z} \neq y_k^i$):
14 Set $\alpha_k^i = 0$, $\tilde{\alpha}_{k+1}^i = \tilde{\alpha}_k^i$, $y_k^{n+1} = \tilde{z}$, $d_k^i = -d_{k-1}^i$ and
15 **Exit For**
16 **Case** L_-^2 ($\alpha = 0$ and $\tilde{z} = y_k^i$):
17 Set $\alpha_k^i = 0$, $\tilde{\alpha}_{k+1}^i = \max\{1, \lfloor \tilde{\alpha}_k^i/2 \rfloor\}$ and $d_k^i = d_{k-1}^i$.
18 **Case** L_-^3 ($\alpha \neq 0$):
19 Set $\alpha_k^i = \alpha$, $\tilde{\alpha}_{k+1}^i = \alpha$ and $d_k^i = -d_{k-1}^i$.
20 **Case** L_+^3 ($\alpha \neq 0$):
21 Set $\alpha_k^i = \alpha$, $\tilde{\alpha}_{k+1}^i = \alpha$ and $d_k^i = d_{k-1}^i$.
22 **Endif**
23 Set $y_k^{i+1} = y_k^i + \alpha_k^i d_k^i$.
24 **End For**
25 **If** $(y_k^i)_z = (x_k^i)_z$ and $\tilde{\alpha}_k^i = 1$, $i \in I_z$, **then**
26 **If** $(\max_{i \in I_c}\{\alpha_k^i, \tilde{\alpha}_k^i\} \leq \epsilon_k^q)$ and $(\|g^+(x_k)\| > \eta_k)$, choose $\epsilon_{k+1} = \theta \epsilon_k$.
27 **Else** set $\epsilon_{k+1} = \epsilon_k$.
28 set $\xi_{k+1} = \theta \xi_k$,
29 **Else** set $\xi_{k+1} = \xi_k$, $\epsilon_{k+1} = \epsilon_k$.
30 Set $\eta_{k+1} = \theta \eta_k$.
31 Find $x_{k+1} \in \mathcal{X} \cap \mathcal{Z}$ such that $P(x_{k+1}; \epsilon_k) \leq P(y_k^{n+1}; \epsilon_k)$.
32 **End For**

Finally, once all directions have been investigated, or when case L_+^1 or L_-^1 is executed, the algorithm jumps directly to step 25, where ξ_{k+1}, ϵ_{k+1}, and η_{k+1} are computed.

In the following, we carry out the convergence analysis of Algorithm EDFL.

Lemma 4.1 *The Local Search procedure is well defined (i.e., it cannot indefinitely cycle between Steps 2 and 4).*

Proof In Lemma 3.1, we already proved that the Continuous and Discrete search procedures are both well defined. Hence, in order to prove that the Local Search procedure is well defined, we need to show that it could not indefinitely cycle between Steps 2 and 4. On the contrary, if this was the case, a sequence $\{\beta_l\}$ would exist such that

$$\lim_{l\to\infty} P(y+\beta_l p;\epsilon) = -\infty,$$

but this would contradict the assumption that set $\mathcal{X}$ is compact. □

Lemma 4.2 *Algorithm EDFL is well defined (i.e., it produces an infinite sequence of iterates $\{x_k\}$).*

Proof The result follows from the fact that the procedures used in Algorithm EDFL (Continuous and Local Search) are well defined. □

Proposition 4.1 *Let $\{x_k\}$ and $\{\epsilon_k\}$ be the sequences produced by Algorithm EDFL. Let $K_\xi \subseteq \{1,2,\ldots\}$ and $K_\epsilon \subseteq K_\xi$ be defined in (9). Then, $\{x_k\}$ admits limit points and*

(i) if $\lim_{k\to\infty}\epsilon_k = \bar{\epsilon}$, then every limit point of $\{x_k\}_{K_\xi}$ is stationary for Problem (1);
(ii) if $\lim_{k\to\infty}\epsilon_k = 0$, then every limit point of $\{x_k\}_{K_\epsilon}$ is stationary for Problem (1).

Proof Since the Local Search procedure is an enrichment of the Discrete search procedure used in the definition of Algorithm DFL, the proof follows easily from Theorem 3.1. □

Proposition 4.2 *Let $\{x_k\}$ and $\{\epsilon_k\}$ be the sequences produced by Algorithm EDFL. Let $K_\xi \subseteq \{1,2,\ldots\}$ and $K_\epsilon \subseteq K_\xi$ be defined in (9). Then, $\{x_k\}$ admits limit points and*

(i) if $\lim_{k\to\infty}\epsilon_k = \bar{\epsilon}$, then every limit point of $\{x_k\}_{K_\xi}$ is extended stationary for Problem (1);
(ii) if $\lim_{k\to\infty}\epsilon_k = 0$, then every limit point of $\{x_k\}_{K_\epsilon}$ is extended stationary for Problem (1).

Proof As in the proof of Proposition 3.2, let us denote

$$\tilde{K} := \begin{cases} K_\xi, \text{ if } \lim_{k\to\infty}\epsilon_k = \bar{\epsilon}, \\ K_\epsilon, \text{ if } \lim_{k\to\infty}\epsilon_k = 0. \end{cases}$$

By the instructions of Algorithm EDFL, every iterate x_k belongs to $\mathcal{X}$, which is compact. Hence $\{x_k\}$ admits limit points.

Let $x^\star$ be a limit point of $\{x_k\}_{\tilde{K}}$ and $\bar{K} \subseteq \tilde{K}$ be an index set such that

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

By recalling the definition of extended Stationary Point, we have to show that a $\lambda^\star \in \mathbb{R}^m$ exists such that the pair $(x^\star, \lambda^\star)$ satisfies (2), (4), and (3). Recalling the fact that the Local Search is an enrichment of the Discrete search defined in Sect. 3, the limit points produced by Algorithm EDFL surely satisfy (2), (4), and (3), which can be derived by using point (i) of Proposition 3.1 and Proposition 3.2.

Now we show that, for all $\bar{x} \in \mathcal{B}_z(x^\star) \cap \mathcal{F}$ such that $f(\bar{x}) = f(x^\star)$, it is possible to find a $\bar{\lambda} \in \mathbb{R}^m$ such that the pair $(\bar{x}, \bar{\lambda})$ satisfies (5), (7), and (6).

First we note that, by Lemmas 3.3 and 3.4, we have

$$\lim_{k\to\infty, k\in\bar{K}} \xi_k = 0, \tag{25a}$$

$$\lim_{k\to\infty, k\in\bar{K}} y_k^i = x^\star, \quad i = 1, \ldots, n+1. \tag{25b}$$

Furthermore, for any choice of $\bar{x} \in \mathcal{B}_z(x^\star) \cap \mathcal{F}$ such that $f(\bar{x}) = f(x^\star)$, and reasoning as in Proposition 3.2, there exist an index $\bar{\imath}$ and a subset of indices, which we relabel again $\bar{K}$, such that $i_k = \bar{\imath}$, and recalling the definition of z in the Local Search procedure:

$$\lim_{k\to\infty, k\in\bar{K}} z_k^{\bar{\imath}} = \lim_{k\to\infty, k\in\bar{K}} y_k^{\bar{\imath}} + d_k^{\bar{\imath}} = x^\star + \bar{d} = \bar{x}. \tag{26}$$

Recalling the expression of the penalty function P and the functions λ_l (defined in (8)), we can write

$$\begin{aligned} P(y_k^{\bar{\imath}}; \epsilon_k) &= f(y_k^{\bar{\imath}}) + \frac{1}{\epsilon_k} \sum_{l=1}^{m} \max\{0, g_l(y_k^{\bar{\imath}})\}^q \\ &= f(y_k^{\bar{\imath}}) + \frac{1}{q} \sum_{l=1}^{m} \lambda_l(y_k^{\bar{\imath}}; \epsilon_k) \max\{0, g_l(y_k^{\bar{\imath}})\}. \end{aligned}$$

Now the proof continues by showing that

(i) the following limits hold

$$\lim_{k\to\infty, k\in\bar{K}} P(y_k^{\bar{\imath}}; \epsilon_k) = f(x^\star), \tag{27}$$

$$\lim_{k\to\infty, k\in\bar{K}} P(z_k^{\bar{\imath}}; \epsilon_k) = f(\bar{x}), \tag{28}$$

$$\lim_{k\to\infty, k\in\bar{K}} P(w_k^i; \epsilon_k) = f(\bar{x}), \qquad \forall\, i = 1, \ldots, n+1; \tag{29}$$

(ii) point $\bar{x}$ is stationary w.r.t. the continuous variables;
(iii) point $\bar{x}$ is a local minimum w.r.t. the discrete variables.

Point (i). By using Proposition 7.1 in Appendix and recalling that $x^\star \in \mathcal{F}$, we have

$$\lim_{k\to\infty, k\in\bar{K}} \lambda_l(y_k^{\bar{\imath}}; \epsilon_k) \max\{0, g_l(y_k^{\bar{\imath}})\} = 0. \tag{30}$$

Therefore, (27) follows.

Now we show that

$$\lim_{k\to\infty, k\in\bar{K}} \lambda_l(z_k^{\bar{l}};\epsilon_k)\max\{0, g_l(z_k^{\bar{l}})\} = 0, \tag{31}$$

for all $l = 1,\ldots,m$.

If part (ii) of Assumption 2.2 holds, then by (26) and the fact that $\bar{x} \in \mathcal{F}$, for sufficiently large $k \in \bar{K}$, we can write

$$\lambda_l(z_k^{\bar{l}};\epsilon_k)\max\{0, g_l(z_k^{\bar{l}})\} = 0.$$

On the other hand, if part (i) of Assumption 2.2 holds, considering that $z_k^{\bar{l}} = y_k^{\bar{l}} + d_k^{\bar{l}}$, $\bar{l} \in I_z$, and recalling the expression (8) of $\lambda_l(x;\epsilon)$, we have

$$\lambda_l(z_k^{\bar{l}};\epsilon_k)\max\{0, g_l(z_k^{\bar{l}})\} = \lambda_l(y_k^{\bar{l}};\epsilon_k)\max\{0, g_l(y_k^{\bar{l}})\}.$$

Thus, relations (31) follow from the above relation and (30).

For every $k \in \bar{K}$, it results that

$$P(z_k^{\bar{l}};\epsilon_k) = P(w_k^1;\epsilon_k) \geq P(w_k^2;\epsilon_k) \geq \cdots \geq P(w_k^n;\epsilon_k) > P(y_k^{\bar{l}};\epsilon_k) - \xi_k.$$

From (26) and (31), we obtain (28). Then, by (25), (27), (28) and by considering that, by assumption, $f(\bar{x}) = f(x^\star)$, we get (29).

Point (ii). For every $i \in I_c$ such that

$$P(w_k^i + \tilde{\alpha}_k^i q_k^i;\epsilon_k) > P(w_k^i;\epsilon_k) - \gamma(\tilde{\alpha}_k^i)^2,$$

we have that $w_k^{i+1} = w_k^i$, and by Lemma 3.2, $\tilde{\alpha}_k^i \to 0$, for all $i \in I_c$.

On the other hand, for those indices $i \in I_c$ such that

$$P(w_k^{i+1};\epsilon_k) = P(w_k^i + \hat{\alpha}_k^i q_k^i;\epsilon_k) \leq P(w_k^i;\epsilon_k) - \gamma(\hat{\alpha}_k^i)^2, \tag{32}$$

we have, by (29) and (32), that

$$\lim_{k\to\infty, k\in\bar{K}} \hat{\alpha}_k^i = 0, \forall\, i \in I_c. \tag{33}$$

Hence, recalling that $w_k^1 = z_k^{\bar{l}}$ by definition of the Local Search procedure, by (26), and the fact that $\tilde{\alpha}_k^i \to 0$ and $\hat{\alpha}_k^i \to 0$, we have that

$$\lim_{k\to\infty, k\in\bar{K}} w_k^i = \bar{x}, \forall\, i \in I_c \tag{34}$$

Now, for k sufficiently large, by Proposition 2.2, $D(\bar{x}) \subseteq D(x_k)$. Since the Grid Search step in the Local search procedure explores, for every index i, both the directions e^i and $-e^i$, for every $i \in I_c$ and $\bar{d}^i \in D(\bar{x})$, we can define η_k^i as follows:

$$\eta_k^i := \begin{cases} \tilde{\alpha}_k^i, & \text{if } P(w_k^i + \tilde{\alpha}_k^i \bar{d}^i; \epsilon_k) > P(w_k^i; \epsilon_k) - \gamma(\tilde{\alpha}_k^i)^2, \\ \dfrac{\hat{\alpha}_k^i}{\delta}, & \text{if } P\left(w_k^i + \dfrac{\hat{\alpha}_k^i}{\delta}\bar{d}^i; \epsilon_k\right) > P(w_k^i; \epsilon_k) - \gamma\left(\dfrac{\hat{\alpha}_k^i}{\delta}\right)^2. \end{cases}$$

Then, we can write

$$P(w_k^i + \eta_k^i \bar{d}^i; \epsilon_k) > P(w_k^i; \epsilon_k) - \gamma(\eta_k^i)^2. \tag{35}$$

By Lemma 3.2, we have, for all $i \in I_c$, that

$$\lim_{k\to\infty, k\in\bar{K}} \eta_k^i = 0. \tag{36}$$

From (26) and (34), it follows that

$$\lim_{k\to\infty, k\in\bar{K}} \|z_k^{\bar{\imath}} - w_k^i\| = 0 \tag{37}$$

for all $i \in I_c$. Then, by (26) and the fact that $\bar{x} \in \mathcal{F}$, we can write

$$\lim_{k\to\infty, k\in\bar{K}} \epsilon_k \|g^+(z_k^{\bar{\imath}})\| = 0. \tag{38}$$

Thus, considering that, for k sufficiently large, $w_k^i + \eta_k^i \bar{d}^i \in \mathcal{X} \cap \mathcal{Z}$, (35), (36), (37), and (38) prove that the hypotheses of Proposition 7.1 in Appendix are satisfied. Hence, $\bar{x}$ is stationary with respect to the continuous variables.

Point (iii). Finally, again reasoning as in the proof of Proposition 3.2, considering (33) and using $w_k^j = z_k^{\bar{\imath}} + \sum_{h=1}^{j} \hat{\alpha}^h q^h$ and $w_k^j + q_k^j$ (omitting the dependence of w_k and q_k from the index $\bar{\imath}$) in place of, respectively, y_k^i and $y_k^i + d_k^i$, we can find an index $\bar{\jmath} \in I_z$ and a subset of indices $\hat{K}$, such that

$$\lim_{k\to\infty, k\in\hat{K}} w_k^{\bar{\jmath}} = \bar{x}, \tag{39}$$

$$\lim_{k\to\infty, k\in\hat{K}} w_k^{\bar{\jmath}} + q_k^{\bar{\jmath}} = \tilde{x}, \tag{40}$$

where $\tilde{x}$ is a point belonging to the discrete neighborhood of $\bar{x}$. Hence, reasoning as in Proposition 3.2, for k sufficiently large and $k \in \hat{K}$,

$$w_k^{\bar{\jmath}} + q_k^{\bar{\jmath}} \in \mathcal{X} \cap \mathcal{Z}.$$

Then, we have

$$P(w_k^{\bar{\jmath}} + q_k^{\bar{\jmath}}; \epsilon_k) > P(y_k^{\bar{\imath}}; \epsilon_k) - \xi_k. \tag{41}$$

Now we show that

$$\lim_{k\to\infty, k\in\bar{K}} \lambda_l(w_k^{\bar{\jmath}} + q_k^{\bar{\jmath}}; \epsilon_k) \max\{0, g_l(w_k^{\bar{\jmath}} + q_k^{\bar{\jmath}})\} = 0, \tag{42}$$

for all $l = 1, \ldots, m$.

First, if part (ii) of Assumption 2.2 holds, then by (26), (39), (40), and the fact that $\tilde{x} \in \mathcal{F}$ and $\bar{x} \in \mathcal{F}$, for sufficiently large $k \in \bar{K}$, we can write

$$\lambda_l(w_k^{\bar{j}} + q_k^{\bar{j}}; \epsilon_k)\max\{0, g_l(w_k^{\bar{j}} + q_k^{\bar{j}})\} = 0.$$

On the other hand, if part (i) of Assumption 2.2 holds, considering that, by (26), (39), and (40),

$$\begin{aligned} &\lim_{k\to\infty, k\in\bar{K}} \lambda_l(w_k^{\bar{j}} + q_k^{\bar{j}}; \epsilon_k)\max\{0, g_l(w_k^{\bar{j}} + q_k^{\bar{j}})\} \\ &= \lim_{k\to\infty, k\in\bar{K}} \lambda_l(z_k^{\bar{l}} + q_k^{\bar{j}}; \epsilon_k)\max\{0, g_l(z_k^{\bar{l}} + q_k^{\bar{j}})\}, \end{aligned} \tag{43}$$

for all $l = 1, \ldots, m$, then we show (42) by proving

$$\lim_{k\to\infty, k\in\bar{K}} \lambda_l(z_k^{\bar{l}} + q_k^{\bar{j}}; \epsilon_k)\max\{0, g_l(z_k^{\bar{l}} + q_k^{\bar{j}})\} = 0,$$

for all $l = 1, \ldots, m$. Indeed, considering that $z_k^{\bar{l}} = y_k^{\bar{l}} + d_k^{\bar{l}}$, $\bar{l} \in I_z$, and recalling the expression (8) of $\lambda_l(x; \epsilon)$, we have

$$\begin{aligned} \lambda_l(z_k^{\bar{l}} + q_k^{\bar{j}}; \epsilon_k)\max\{0, g_l(z_k^{\bar{l}} + q_k^{\bar{j}})\} &= \lambda_l(z_k^{\bar{l}}; \epsilon_k)\max\{0, g_l(z_k^{\bar{l}})\} \\ &= \lambda_l(y_k^{\bar{l}} + d_k^{\bar{l}}; \epsilon_k)\max\{0, g_l(y_k^{\bar{l}} + d_k^{\bar{l}})\} \\ &= \lambda_l(y_k^{\bar{l}}; \epsilon_k)\max\{0, g_l(y_k^{\bar{l}})\}. \end{aligned}$$

Thus, (42) follows from the above relation and (30).

Hence, by (40) and (42), we can write

$$\lim_{k\to\infty, k\in\bar{K}} P(w_k^{\bar{j}} + q_k^{\bar{j}}; \epsilon_k) = f(\tilde{x}).$$

Now, recalling (25) and (26), relation (6) follows by taking the limit for $k \to \infty$, $k \in \bar{K}$ in (41), and considering that, by assumption, $f(\bar{x}) = f(x^\star)$. □

5 Numerical Results

In this section, we report the numerical performance of the proposed derivative-free algorithms DFL[1] and EDFL[1] for MINLP (see footnote 1) problems both on a set of academic test problems and on a real application arising in the optimal design of industrial electric motors. Moreover, a comparison with NOMAD v3.6.0, which is a well-known software package for derivative-free optimization [26], on the same set of test problems and on the real problem is carried out. It is worth noting that the MADS

[1] Available for download at: http://www.dis.uniroma1.it/~lucidi/DFL.

algorithm implemented in NOMAD is designed only for continuous and categorical variables, but can be adapted to take into account the presence of discrete variables. Theory about MADS with discrete variables can be found in [2].

The proposed method has been implemented in double precision Fortran90 and all the experiments have been conducted by choosing the following values for the parameters defining Algorithm DFL: $\gamma = 10^{-6}$, $\theta = 0.5$, $p = 2$,

$$\tilde{\alpha}_0^i := \begin{cases} \max\left\{10^{-3}, \min\{1, |(x_0)^i|\}\right\}, & i \in I_c, \\ \max\left\{1, \min\{2, |(x_0)^i|\}\right\}, & i \in I_z. \end{cases}$$

As concerns the penalty parameter, in the implementation of Algorithm DFL, we use a vector of penalty parameters $\epsilon \in \mathbb{R}^m$ and choose

$$(\epsilon_0)^j := \begin{cases} 10^{-3}, \text{ if } g_j(x_0)^+ < 1, \\ 10^{-1}, \text{ otherwise.} \end{cases} \quad j = 1, \ldots, m. \tag{44}$$

In order to preserve all the theoretical results, the test at step 14 of Algorithm DFL and at step 26 of Algorithm EDFL $\max_{i=1,\ldots,n}\{\tilde{\alpha}_k^i, \alpha_k^i\} \leq \epsilon_k^p$ has been substituted by

$$\max_{i=1,\ldots,n} \{\tilde{\alpha}_k^i, \alpha_k^i\} \leq \max_{i=1,\ldots,m} \{(\epsilon_k)^i\}^p.$$

The parameters defining Algorithm EDFL have been set to the same values used in DFL except for the new parameter ν of the Local search procedure which is set equal to 1.

5.1 Results on Test Problems

We selected a set of 50 test problems from the well-known collections [27,28] which have been suitably modified by letting some variables assume only a finite number of values. In particular, for every even index i, variable $x^i \in \mathcal{X}^i$ with

$$\mathcal{X}^i := \left\{ l^i + h\frac{(u^i - l^i)}{20} \right\} \quad \text{for } h = 0, \ldots, 20.$$

In Table 1, we report the details of the selected test problems. Namely, for each problem, we indicate by n, the number of variables, and by m, the number of nonlinear plus general linear constraints; f_0 denotes the value of the objective function on the initial point, that is, $f_0 = f(x_0)$; finally, viol_0 is a measure of the infeasibility on the initial point, that is, $\text{viol}_0 = \sum_{j=1}^m \max\{0, g_j(x_0)\}$. In the table, we evidenced (by a '$*$' symbol after the name) the problems whose initial points are infeasible with respect to the bound constraints. In those cases, we obtained an initial point by projecting the provided point onto the set defined by the bound constraints.

As concerns NOMAD, we first run the package by using default values for all of the parameters. Then, we run a modified version of NOMAD, namely NOMAD*,

Table 1 Test problems characteristics

Problem	n	m	f_0	Viol_0
HS 14	2	3	1.00E+00	3.000E+00
HS 15	2	2	1.61E+03	3.000E+00
HS 16	2	2	5.85E+01	0.000E+00
HS 18	2	2	6.25E+02	0.000E+00
HS 19	2	2	2.80E+04	2.141E+03
HS 20	2	3	8.50E+00	1.250E+00
HS 21	2	1	−1.00E+02	0.000E+00
HS 22	2	2	1.00E+00	4.000E+00
HS 23	2	5	9.00E+00	3.000E+00
HS 30	3	1	2.00E+00	0.000E+00
HS 31	3	1	4.82E+01	0.000E+00
HS 39	4	4	−2.00E+00	1.600E+01
HS 40	4	6	−0.00E+00	1.288E+00
HS 42	4	4	2.40E+01	2.000E+00
HS 43	4	3	0.00E+00	0.000E+00
HS 60	3	2	2.10E+01	9.757E+00
HS 64	3	1	7.20E+09	3.200E+06
HS 65	3	1	6.74E+01	0.000E+00
HS 72	4	2	2.00E+05	5.750E+00
HS 74	4	8	1.34E+03	1.400E+03
HS 75	4	8	1.34E+03	1.400E+03
HS 78	5	6	0.00E+00	8.000E+00
HS 79	5	6	4.10E+01	1.059E+01
HS 80	5	6	1.00E+00	8.000E+00
HS 83	5	6	−3.22E+04	2.773E+00
HS 95	6	4	1.09E+00	9.495E+01
HS 96	6	4	1.09E+00	1.749E+02
HS 97	6	4	1.09E+00	9.495E+01
HS 98	6	4	1.09E+00	2.849E+02
HS 100	7	4	1.16E+03	0.000E+00
HS 101	7	6	2.21E+03	3.703E+02
HS 102	7	6	2.21E+03	3.703E+02
HS 103	7	6	2.21E+03	3.703E+02
HS 104	8	6	6.79E−01	5.187E+00
HS 106	8	6	1.55E+04	1.346E+06
HS 107	9	6	2.91E+03	1.349E+00
HS 108	9	13	−0.00E+00	3.000E+00
HS 113	10	8	1.32E+03	9.300E+01
HS 114	10	14	2.19E+03	1.178E+03
HS 116	13	15	2.25E+02	7.902E+02

Table 1 continued

Problem	n	m	f_0	Viol_0
HS 223	2	2	−1.00E−01	0.000E+00
HS 225	2	5	9.00E+00	3.000E+00
HS 228	2	2	0.00E+00	0.000E+00
HS 230	2	2	0.00E+00	1.000E+00
HS 263	4	6	−1.00E+01	2.200E+03
HS 315	2	3	−0.00E+00	0.000E+00
HS 323	2	2	4.00E+00	1.000E+00
HS 343	3	2	−1.94E+01	5.686E+02
HS 365∗	7	5	6.00E+00	7.341E+00
HS 369∗	8	6	6.60E+03	5.449E+01
HS 372∗	9	12	3.59E+05	5.610E+02
HS 373	9	12	3.82E+05	1.014E+03
HS 374	10	35	0.00E+00	9.612E+00

in which we set the `MODEL_SEARCH` parameter to `NO`, thus disabling the NOMAD search strategy using quadratic models.

We give all the solvers a maximum of 1300 function evaluations (i.e., the equivalent of 100 simplex gradient evaluations for a problem with $n = 13$ variables, like our biggest test problem).

In Fig. 2, we report the comparison between NOMAD, NOMAD*, DFL, and EDFL in terms of performance and data profiles.

In order to adapt the procedure for constructing performance and data profiles, as proposed in [29], to the nonlinearly constrained case, we considered the convergence test

$$\tilde{f}_0 - f(x) \geq (1-\tau)(\tilde{f}_0 - f_L),$$

where $\tilde{f}_0$ is the objective function value of the *worst feasible point* determined by all the solvers, $\tau > 0$ is a tolerance, and f_L is computed for each problem as the smallest value of f (at a feasible point) obtained by any solver within the allowed 1300 function evaluations. We note that, when a point is not feasible (i.e., $\text{viol}(x) = \sum_{j=1}^{m} \max\{0, g_j(x)\} > 10^{-6}$), we set $f(x) = +\infty$.

The results reported in Fig. 2 show that NOMAD and EDFL are slightly the best solvers for $\tau = 10^{-3}$, whereas for $\tau = 10^{-1}$, DFL outperforms NOMAD. We believe that the performances of both DFL and EDFL could be further improved by introducing in the latter algorithms the use of quadratic models to (possibly) improve the current iterate.

5.2 Results on an Optimal Design Problem

In this section, we report the results obtained by the three codes (DFL, EDFL, and NOMAD) on a real optimal design problem. In DFL and EDFL, we use as stopping

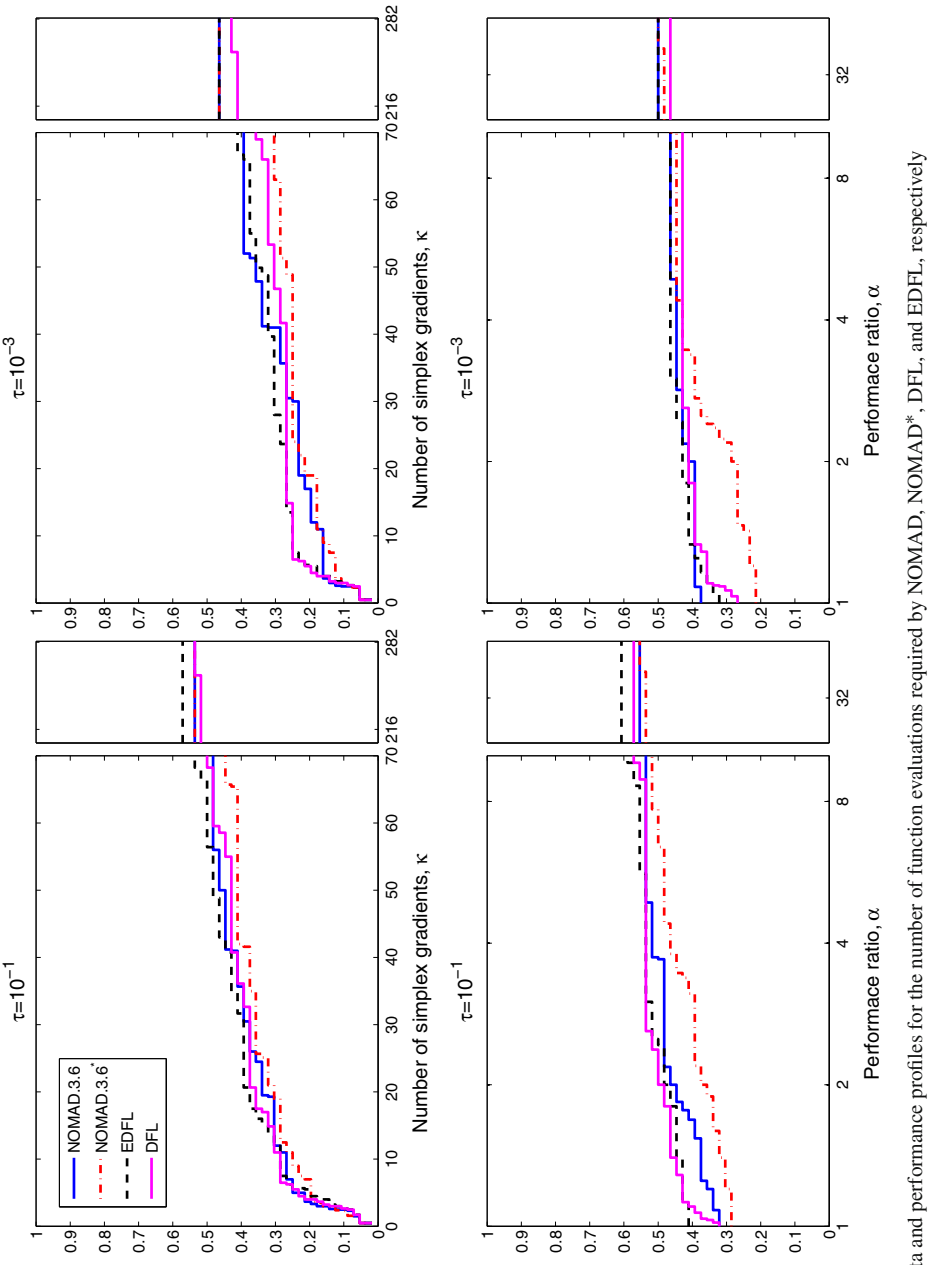

Fig. 2 Data and performance profiles for the number of function evaluations required by NOMAD, NOMAD*, DFL, and EDFL, respectively

Table 2 Lower and upper bounds of continuous design variables

	Meaning	l.b.	u.b.
Continuous variables			
x_3	Inner stator diameter (mm)	72	80
x_4	Stator tooth width (mm)	2.5	3.5
x_5	Stator yoke thickness (mm)	4.0	8.0
x_6	Slot opening width (mm)	1.2	1.6
x_7	Slot opening depth (mm)	1.0	2.0
x_8	Bottom loop radius (mm)	0.3	0.8
x_9	Upper loop radius (mm)	0.3	0.8
x_{10}	PM thickness (mm)	2.0	4.0
x_{11}	Ratio of PM width to barrier width	0.80	0.95
x_{12}	Magnet position (mm)	4.0	8.0
x_{13}	Rotor tooth width (mm)	4.0	6.0

condition $\max_{i\in I_c}\{\tilde{\alpha}_k^i, \alpha_k^i\} \leq 10^{-6}$. We note that, as a consequence of this stopping condition and of the initialization (44), the final values of the penalty parameters are greater than 10^{-6}. As for NOMAD, we set the parameter `MIN_MESH_SIZE` $= 10^{-6}$.

We consider the optimal design of Interior Permanent Magnet (IPM) synchronous motors [30] which are built with magnets placed inside the rotor body and are attracting great attention in several variable speed applications, such as electric vehicles, industrial, and domestic appliances. The most challenging requirements are, among others, high torque at base and maximum speed, limited gross weight, and extended speed range.

In Tables 2 and 3, we report the meaning of the optimization variables along with their upper (u.b.) and lower (l.b.) bounds. For discrete variables, in Table 3, we also specify the allowed step. Figure 3 depicts a cross section of one pole of the considered motor and the related design variables. Table 4 reports the nonlinear constraints considered during the optimization and their imposed bounds. Finally, we mention that the objective function employed is given by the following expression:

$$f(x) = f_1(x) - f_2(x) - f_3(x),$$

where $f_1(x)$ is the gross weight of the motor (to be minimized), $f_2(x)$ is the torque at base speed (to be maximized), and $f_3(x)$ is the torque at maximum speed (to be maximized).

We remark that all of the constraints and objective function nonlinearly depend on the design variables. Furthermore, since their values are computed by means of a finite element simulation program (which takes about three minutes for each evaluation), they are black-box-type functions whose expressions and first-order derivatives are not known.

We preliminary tried to solve the design problem by using a naive approach. More in particular, we run our DF algorithm relaxing the integrality constraint on the design variables. This produces solution $\bar{x}$ with $f(\bar{x}) = -11.006$, which is infeasible because of the nonintegrality of variables x_{14} and x_{15}. Then, we rounded $\bar{x}_{14}$ and $\bar{x}_{15}$ to the

Table 3 Lower and upper bounds of discrete design variables

	Meaning	l.b.	u.b.	Step
Discrete variables				
x_1	Stack length (mm)	60	90	1
x_2	Outer stator diameter (mm)	105	130	1
x_{14}	Angle of flux barrier (°)	−10	10	1
x_{15}	Angle of flux barrier (°)	−10	10	1
x_{16}	Number of wires per slot	4	14	1
x_{17}	Wire size (mm)	1.0	3.0	0.01

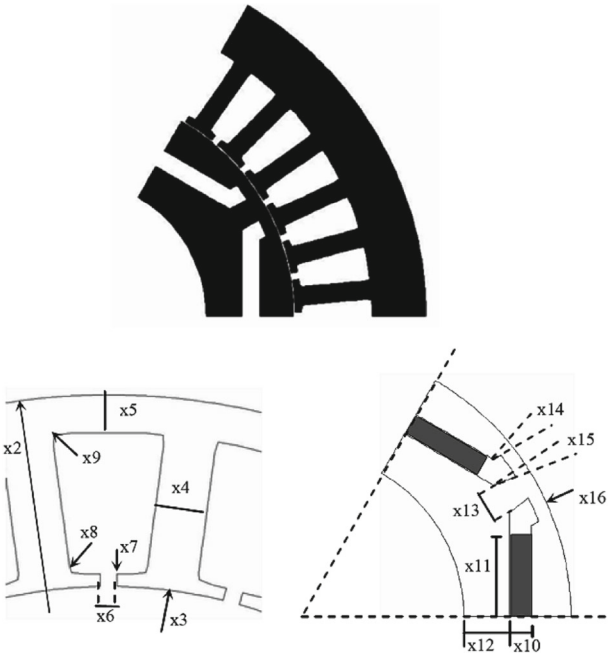

Fig. 3 Cross section of the considered IPM motor (one pole) and design variables, except for the wire size (×17)

nearest integer, and thus obtaining a point $\tilde{x}$ which violates a nonlinear constraint, namely $g_7(x)$. Hence, in order to recover feasibility, we run our method by holding that fixed the discrete variables to their rounded values. This indeed allows to recover feasibility and produces a point x^* with $f(x^*) = -11.2524$.

In Tables 5 and 6, we summarize the results obtained in terms of black-box evaluations (nf), objective functions values, and computed solution points.

As concerns the performance of the codes on this real problem, DFL requires 585 function evaluations to satisfy the stopping condition (i.e., $\max_{i \in I_c}\{\tilde{\alpha}_k^i, \alpha_k^i\} \leq 10^{-6}$). The final point $x^\star$, obtained by DFL, has $f(x^\star) = -12.1631$ and is feasible. On the

Table 4 Bounds on the nonlinear constraints

	Meaning	Bound
Nonlinear constraints		
g1	Stator slot fill factor (%)	≤ 40
g2	Max flux density in the stator tooth (T)	≤ 1.83
g3	Max flux density in the stator yoke (T)	≤ 1.80
g4	Linear current density (A/cm)	≤ 400
g5	Maximum speed (rpm)	$\geq 40{,}000$
g6	Back EMF at maximum speed [V]	≤ 210
g7	Phase resistance at 90° (Ω)	≤ 0.31
g8	Torque at base speed (Nm)	≥ 9.5
g9	Torque at maximum speed (Nm)	≥ 1.8
g10	Gross weight (kg)	≤ 7.5

Table 5 Summary of the results obtained for the IPM motor design

	nf	$f_1(x)$	$f_2(x)$	$f_3(x)$	$f(x)$
Naive	833	5.93	13.8175	3.3649	−11.2524
DFL	585	6.6713	15.0377	3.7967	−12.1631
EDFL	2483	6.9161	15.6321	3.7769	−12.4929
NOMAD	3267	7.0857	12.9526	2.9023	−8.7692

Table 6 Solution points obtained by the naive approach, DFL, EDFL, and NOMAD

	Naive	DFL	EDFL	NOMAD
x_1	80	90	90	89
x_2	110	110	112	114
x_3	80	80	80	79.988
x_4	2.0	2.0	2.016	2.0
x_5	5.0	5.0	5.0	5.0
x_6	1.6	1.6	1.6	1.256
x_7	1.004	1.111	2.0	2.0
x_8	0.8	0.8	0.8	0.3
x_9	0.8	0.8	0.3	0.8
x_{10}	2.712	2.611	2.643	3.5
x_{11}	0.95	0.9	0.9	0.93
x_{12}	4.25	4.02	4.414	4.004
x_{13}	4.937	4.0	4.5	4.801
x_{14}	−2	2	0	−4
x_{15}	2	0	1	10
x_{16}	10	10	10	8
x_{17}	1.8	1.8	1.8	2.17

other hand, EDFL requires 2483 function evaluations to satisfy the stopping condition. It gives a final solution $x^\star$ with $f(x^\star) = -12.4929$ which is feasible.

Then, we ran NOMAD on the optimal design problem by using the same parameter settings as those used for test problems experimentation. NOMAD stopped after 3267 black-box function evaluations reporting a best feasible point $x^\star$ with $f(x^\star) = -8.7692$.

Finally, it is worth noting that the three competing algorithms require a different effort, in term of function evaluations, to find a first feasible solution. In particular, DFL, EDFL, and NOMAD require 297, 1071, and 1216, respectively.

6 Conclusions

In this paper, we have addressed MINLP problems. First, we have introduced the definition of stationary and extended stationary points, and then we have proposed two algorithms and proved their global convergence. The first algorithm, namely DFL, converges toward stationary points whereas the second algorithm, EDFL, converges toward extended stationary points. The proposed algorithms are of the linesearch type, in the sense that, along the continuous variables, we adopt a well-studied linesearch with sufficient decrease strategy. The two algorithms differ in the way the discrete variables are updated. DFL manages the discrete variables by means of a Discrete search procedure whereas EDFL performs a deeper investigation of the discrete neighborhood by using a Local Search procedure, which is a Discrete search enriched by a so-called Grid search phase. All the methods proposed use a sequential penalty approach to tackle general nonlinear constraints, and thus forcing feasibility of the iterates in the limit as the penalty parameter goes to zero.

The two algorithms have been tested both on a modified set of known test problems and on a real optimal design problem, and their performances have been compared with those of the well-known derivative-free optimization package NOMAD. The numerical experimentation proves the efficiency of the proposed methods and shows that EDFL is able to find better solutions than DFL at the cost of a higher number of function evaluations.

Acknowledgments We are indebted to two anonymous Reviewers whose many interesting suggestions and stimulating comments greatly helped us improving the paper. We thank Professors Franco Parasiliti and Marco Villani for providing us the optimal design problem and for giving useful insights and comments on the obtained results. Finally, we would like to mention that this work has been partially funded by the UE (ENIAC Joint Undertaking) in the MODERN project (ENIAC-120003).

Appendix: Technical Result

In this section, we report a technical result which is needed to prove convergence of DFL and EDFL. It is a slight modification of an analogous result reported in [20], which takes into account the presence of discrete variables.

Proposition 7.1 *Let $\{\epsilon_k\}$ be a bounded sequence of positive penalty parameters. Let $\{x_k\}$ be a sequence of points such that $x_k \in \mathcal{X} \cap \mathcal{Z}$ for all k, and let $\bar{x} \in \mathcal{X} \cap \mathcal{Z}$ be a*

limit point of a subsequence $\{x_k\}_K$ *for some infinite set* $K \subseteq \{0, 1, \ldots\}$. *Suppose that for each* $k \in K$ *sufficiently large,*

(i) for all $d^i \in D^c \cap D(\bar{x})$, *there exist vectors* y_k^i *and scalars* $\eta_k^i > 0$ *such that*

$$y_k^i + \eta_k^i d^i \in \mathcal{X} \cap \mathcal{Z},$$
$$P(y_k^i + \eta_k^i d^i; \epsilon_k) \geq P(y_k^i; \epsilon_k) - o(\eta_k^i),$$
$$\lim_{k\to\infty, k\in K} \frac{\max_{d^i \in D_c \cap D(\bar{x})}\{\eta_k^i, \|x_k - y_k^i\|\}}{\epsilon_k} = 0;$$

(ii)

$$\lim_{k\to\infty, k\in K} \epsilon_k \|g^+(x_k)\| = 0;$$

(iii)

$$(y_k^i)_z = (x_k)_z, \qquad \textit{for all } i \in I_c.$$

Then, $\bar{x}$ *is a stationary point for Problem* (1) *with respect to the continuous variables, that is,* $\bar{x}$ *satisfies* (2) *and* (4) *with* $\bar{\lambda} \in \mathbb{R}^m$ *given by*

$$\lim_{k\to\infty, k\in K} \lambda_j(x_k; \epsilon_k) = \lim_{k\to\infty, k\in K} \lambda_j(y_k^i; \epsilon_k) = \bar{\lambda}_j, \qquad \forall\, i \in I_c \textit{ and } j = 1, \ldots, m,$$

where $\lambda_j(x; \epsilon)$, $j = 1, \ldots, m$, *are defined in* (8).

Proof Considering point (iii), namely that the discrete variables are held fixed in the considered subsequence, the proof is the same as that of Proposition 3.1 in [20]. □

References

1. Conn, A., Scheinberg, K., Vicente, L.: Introduction to Derivative-Free Optimization. MOS/SIAM Series on Optimization. SIAM, Philadelphia, MA (2009)
2. Abramson, M., Audet, C., Chrissis, J., Walston, J.: Mesh adaptive direct search algorithms for mixed variable optimization. Optim. Lett. **3**(1), 35–47 (2009)
3. Audet, C., Dennis Jr, J.: Mesh adaptive direct search algorithms for constrained optimization. SIAM J. Optim. **17**(1), 188–217 (2006)
4. Torczon, V.: On the convergence of pattern search algorithms. SIAM J. Optim. **7**(1), 1–25 (1997)
5. Audet, C., Dennis Jr, J.: Analysis of generalized pattern searches. SIAM J. Optim. **13**(3), 889–903 (2003)
6. Kolda, T., Lewis, R., Torczon, V.: Optimization by direct search: new perspectives on some classical and modern methods. SIAM Rev. **45**(3), 385–482 (2003)
7. Audet, C., Dennis Jr, J.: Pattern search algorithms for mixed variable programming. SIAM J. Optim. **11**(3), 573–594 (2001)
8. Kokkolaras, M., Audet, C., Dennis Jr, J.: Mixed variable optimization of the number and composition of heat intercepts in a thermal insulation system. Optim. Eng. **2**(1), 5–29 (2001)
9. Abramson, M.: Pattern search filter algorithms for mixed variable general constrained optimization problems. Ph.D. thesis, Department of Computational and Applied Mathematics, Rice University (2002)

10. Abramson, M., Audet, C., Dennis Jr, J.: Filter pattern search algorithms for mixed variable constrained optimization problems. Pac. J. Optim. **3**(3), 477–500 (2007)
11. Audet, C., Dennis Jr, J.: A pattern search filter method for nonlinear programming without derivatives. SIAM J. Optim. **14**(4), 980–1010 (2004)
12. Vicente, L.: Implicitly and densely discrete black-box optimization problems. Optim. Lett. **3**, 475–482 (2009)
13. Lucidi, S., Piccialli, V., Sciandrone, M.: An algorithm model for mixed variable programming. SIAM J. Optim. **14**, 1057–1084 (2005)
14. Lucidi, S., Sciandrone, M., Tseng, P.: Objective-derivative-free methods for constrained optimization. Math. Program. **92**(1), 37–59 (2002)
15. García-Palomares, U., Costa-Montenegro, E., Asorey-Cacheda, R., Gonzalez-Castano, F.J.: Adapting derivative free optimization methods to engineering models with discrete variables. Optim. Eng. **13**, 579–594 (2012)
16. García-Palomares, U., Rodríguez, J.: New sequential and parallel derivative-free algorithms for unconstrained optimization. SIAM J. Optim. **13**(1), 79–96 (2002)
17. Müller, J., Shoemaker, C., Piché, R.: So-mi: a surrogate model algorithm for computationally expensive nonlinear mixed-integer black-box global optimization problems. Comput. Oper. Res. **40**(5), 1383–1400 (2013)
18. Laguna, M., Gortázar, F., Gallego, M., Duarte, A., Martí, R.: A black-box scatter search for optimization problems with integer variables. J. Glob. Optim. (2013). doi:10.1007/s10898-013-0061-2
19. Liuzzi, G., Lucidi, S., Rinaldi, F.: Derivative-free methods for bound constrained mixed-integer optimization. Comput. Optim. Appl. **53**, 505–526 (2012)
20. Liuzzi, G., Lucidi, S., Sciandrone, M.: Sequential penalty derivative-free methods for nonlinear constrained optimization. SIAM J. Optim. **20**, 2814–2835 (2010)
21. Lucidi, S., Sciandrone, M.: A derivative-free algorithm for bound constrained optimization. Comput. Optim. Appl. **21**(2), 119–142 (2002)
22. Vicente, L.: Worst case complexity of direct search. EURO J. Comput. Optim. **1**, 143–153 (2013)
23. Nesterov, Y.: Introductory Lectures on Convex Optimization. Kluwer Academic Publishers, Dordrecht (2004)
24. Lin, C.J., Lucidi, S., Palagi, L., Risi, A., Sciandrone, M.: A decomposition algorithm model for singly linearly constrained problems subject to lower and upper bounds. J. Optim. Theory Appl. **141**, 107–126 (2009)
25. Bertsekas, D.: Nonlinear Programming, 2nd edn. Athena Scientific, Belmont, MA (1999)
26. Le Digabel, S.: Algorithm 909: Nomad: nonlinear optimization with the mads algorithm. ACM Trans. Math. Softw. **37**(4), 1–15 (2011)
27. Hock, W., Schittkowski, K.: Test Examples for Nonlinear Programming Codes. Lecture notes in Economics and Mathematical Systems. Springer, Berlin (1981)
28. Hock, W., Schittkowski, K.: More Test Examples for Nonlinear Programming Codes. Lecture notes in Economics and Mathematical Systems. Springer, Berlin (1987)
29. Moré, J., Wild, S.: Benchmarking derivative-free optimization algorithms. SIAM J. Optim. **20**(1), 172–191 (2009)
30. Lucidi, S., Parasiliti, F., Rinaldi, F., Villani, M.: Finite element based multi-objective design optimization procedure of interior permanent magnet synchronous motors for wide constant-power region operation. IEEE Trans. Ind. Electron. **59**(6), 2503–2514 (2012)