ISTITUTO DI ANALISI DEI SISTEMI ED INFORMATICA
"Antonio Ruberti"
CONSIGLIO NAZIONALE DELLE RICERCHE

G. Liuzzi, S. Lucidi, F- Rinaldi

DERIVATIVE-FREE METHODS FOR CONSTRAINED MIXED-INTEGER OPTIMIZATION

R. 11-11, 2011

G. Liuzzi – Consiglio Nazionale delle Ricerche, Istituto di Analisi dei Sistemi ed Informatica "A. Ruberti", Viale Manzoni 30, 00185 Rome, Italy. liuzzi@iasi.cnr.it.

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

F. Rinaldi – "Sapienza" Università di Roma, Dipartimento di Informatica e Sistemistica "A. Ruberti", Via Ariosto 25, 00185 Roma, Italy. rinaldi@dis.uniroma1.it.

This work has been partially funded by the UE (ENIAC Joint Undertaking) in the MODERN project (ENIAC-120003).

ISSN: 1128–3378

Collana dei Rapporti dell'Istituto di Analisi dei Sistemi ed Informatica "Antonio Ruberti", CNR

viale Manzoni 30, 00185 ROMA, Italy

tel. ++39-06-77161
fax ++39-06-7716461
email: iasi@iasi.cnr.it
URL: http://www.iasi.cnr.it

Abstract

We consider the problem of minimizing a continuously differentiable function of several variables subject to simple bound and general nonlinear inequality constraints, where some of the variables are restricted to take integer values. We assume that the first order derivatives of the objective and constraint functions can be neither calculated nor approximated explicitly. This class of mixed integer nonlinear optimization problems arises frequently in many industrial and scientific applications and this motivates the increasing interest in the study of derivative-free methods for their solution. The continuous variables are handled by a linesearch strategy whereas to tackle the discrete ones we employ a local search-type approach. Nonlinear constraints are handled by using a quadratic penalty function approach. We propose two algorithms which are characterized by the way the current iterate is updated and by the stationarity conditions satisfied by the limit points of the sequences they produce. We report a computational experience both on standard test problems and on a real optimal design problem. We also compare the performances of the proposed methods with those of a well-known derivative-free optimization software package, i.e. NOMAD.

Key words: Mixed-integer nonlinear programming, derivative-free optimization, nonlinear constrained optimization.

1. Introduction

In the paper we consider the following Mixed Integer Nonlinear Programming (MINLP) problem

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

where $x \in R^n$, $l, u \in R^n$, and $I_z \subseteq \{1, \ldots, n\}$. We assume $l_i < u_i$, for all $i = 1, \ldots, n$, $l_i, u_i \in Z$, for all $i \in I_z$ and $f : R^n \to R$ and $g_j : R^n \to R$, $j = 1, \ldots, m$, to be $m + 1$ continuously differentiable functions with respect to x_i, $i \notin I_z$. We define the following sets,

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

Further, we introduce the set of indices of continuous variables

$$I_c = \{1, \ldots, n\} \setminus I_z$$

and assume

$$I_c \neq \emptyset.$$

Given a vector $v \in R^n$ we introduce the following subvectors $v_c \in R^{|I_c|}$ and $v_z \in R^{|I_z|}$ given by

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

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

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

We introduce the following definition of neighborhoods with respect to continuous and discrete variables. Given a point $\bar{x} \in R^n$, let us define

$$\begin{aligned} \mathcal{B}_c(\bar{x}, \rho) &= \{x \in R^n : x_z = \bar{x}_z, \ \|x_c - \bar{x}_c\|_2 \leq \rho\}, \\ \mathcal{N}_z(\bar{x}) &= \{x \in R^n : x_c = \bar{x}_c, \ \|x_z - \bar{x}_z\|_2 = 1\}. \end{aligned}$$

In [1, 2, 5, 12] a problem more general than (1) has been considered by allowing also for the presence of categorical variables. The algorithms proposed in the papers [1, 2, 5, 12] are based on the idea of alternating between a local minimization with respect to the continuous variables and a local search with respect to the discrete variables. The common feature of the methods is represented by the fact that the discrete neighborhood structure (that is needed to define the local search) is fixed a priori at every iterate. The cited papers substantially differ in the definition of the continuous minimization phase and in the way the constraints are handled. In [2] a pattern search strategy combined with a filter approach [4] to tackle general nonlinear constraints has been proposed. In [1] for the general nonlinear constrained problem an extreme barrier penalty is adopted and a mesh adaptive direct search strategy [3] is used to force convergence. In [5] the continuous minimization phase is carried out by a pattern search strategy for box constrained problems [8, 16]. Whereas, in [12] a linesearch strategy for linearly constrained problems [15] is adopted. The methods proposed in [5] and [12] are characterized by how they manage feasibility with respect to general constraints. In particular, feasibility is forced explicitly throughout the optimization by an appropriate definition of discrete neighborhoods so that all the iterate they generate are feasible.

To conclude we cite the paper [17] 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 R^n). In [17] a modification of a direct-search algorithm is presented to tackle this kind of problems and a theoretical analysis is reported.

In this paper we extend the approach proposed in [9] for box constrained mixed integer problems by using a sequential quadratic penalty approach like that proposed in [10]. For the continuous variables we adopt a well-studied linesearch with sufficient decrease strategy [13]. For the discrete variables we propose the use of different local search procedures. They explore a discrete neighborhood of points whose structure is not defined a priori but it is adaptively determined by a linesearch-type procedure.

The paper is organized as follows. In Section 2 we introduce some definitions and relevant notations. 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 Problem (1). A computational experience of the methods proposed and comparison with NOMAD is reported in Section 5 both on test problems and on a real optimal design problem. Finally, in Section 6 we draw some conclusions and discuss future developments.

2. Definitions and notations

We begin this section by introducing the following sets of directions.

$$D = \{\pm e_1, \ldots, \pm e_n\}, \qquad D^c = \{\pm e_i : \quad i \in I_c\}, \qquad D^z = \{\pm e_i : \quad i \in I_z\}$$

where e_i, $i = 1, \ldots, n$, is the unit coordinate vector.

Given $x \in X$, 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 $x \in X$, let

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

We report two technical propositions whose proofs can be found, respectively, in [10] and [6].

Proposition 2.1. *For every $x \in X$, it results*

$$cone\{D \cap D(x)\} = D(x). \tag{2}$$

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

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

Due to the mixed-integer nature of Problem (1), different definitions of a local minimum point can be envisaged. In this paper, we consider the following definition of minimum points for Problem (1).

Definition 2.3 (Local minimum point) *A point $x^\star \in \mathcal{F}$ is a local minimum of Problem (1) if, for some $\epsilon > 0$,*

$$\begin{aligned} f(x^\star) \leq f(x), \qquad & \forall x \in \mathcal{B}_c(x^\star; \epsilon) \cap \mathcal{F}, \\ f(x^\star) \leq f(x), \qquad & \forall x \in \mathcal{N}_z(x^\star) \cap \mathcal{F}, \end{aligned} \tag{3}$$

and, every point $\bar{x} \in \mathcal{N}_z(x^\star) \cap \mathcal{F}$ such that $f(\bar{x}) = f(x^\star)$ satisfies (3) for some $\bar{\epsilon} > 0$.

Let us introduce the Lagrangian function associated to Problem (1), that is

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

Proposition 2.4. *Let $x^\star \in \mathcal{F} \cap \mathcal{Z}$ be a local minimum of the problem (1). Then there exists a vector $\lambda^\star \in R^m$ such that*

$$\nabla_c L(x^\star, \lambda^\star)^T (x - x^\star)_c \geq 0, \qquad \forall\ x \in X \tag{4}$$

$$(\lambda^\star)^T g(x^\star) = 0 \qquad \lambda^\star \geq 0 \tag{5}$$

$$f(x^\star) \leq f(x) \qquad \forall\ x \in \mathcal{N}_z(x^\star) \cap \mathcal{F}. \tag{6}$$

Further, for every point $\bar{x} \in \mathcal{N}_z(x^\star) \cap \mathcal{F}$ such that $f(\bar{x}) = f(x^\star)$ a $\bar{\lambda} \in R^m$ exists such that the pair $(\bar{x}, \bar{\lambda})$ satisfies (4), (5) and (6).

Proposition 2.4 essentially states that a minimum point of Problem (1) has to be stationary with respect to the continuous variables and, with respect to the discrete variables, it must be a local minimum within the discrete neighborhood $\mathcal{N}_z(x^\star)$. The same requirement must hold also for those points $\bar{x} \in \mathcal{N}_z(x^\star) \cap \mathcal{F}$ such that $f(\bar{x}) = f(x^\star)$.

With reference to Problem (1), we introduce the following definitions of stationary point and strong stationary point.

Definition 2.5 (Stationary point) *A point $x^\star \in \mathcal{F} \cap \mathcal{Z}$ is a stationary point of Problem (1) if a vector $\lambda^\star \in R^m$ exists such that the pair $(x^\star, \lambda^\star)$ satisfies (4), (5) and (6).*

Definition 2.6 (Strong Stationary Point) *A point $x^\star \in \mathcal{F} \cap \mathcal{Z}$ is a strong stationary point of Problem (1) if a vector $\lambda^\star \in R^m$ exists such that the pair $(x^\star, \lambda^\star)$ satisfies (4), (5) and (6), and, for all $\bar{x} \in \mathcal{N}_z(x^\star) \cap \mathcal{F}$ such that $f(\bar{x}) = f(x^\star)$, it is possible to find a $\bar{\lambda} \in R^m$ so that*

$$\nabla_c L(\bar{x}, \bar{\lambda})^T (x - \bar{x})_c \geq 0, \qquad \forall\ x \in X \tag{7}$$

$$(\bar{\lambda})^T g(\bar{x}) = 0 \qquad \bar{\lambda} \geq 0 \tag{8}$$

$$f(\bar{x}) \leq f(x) \qquad \forall\ x \in \mathcal{N}_z(\bar{x}) \cap \mathcal{F}. \tag{9}$$

In the paper we consider the following penalty function

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

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

$$\lambda_j(x; \epsilon) = \frac{q}{\epsilon} \max\{0, g_j(x)\}^{q-1}, \qquad \forall\ j = 1, \ldots, m. \tag{10}$$

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

Assumption 2.7. *The set X is compact.*

Assumption 2.8. *For every $x \in X$ there exists a vector $\hat{d} \in D(x)$ such that $\hat{d}_i = 0$, for all $i \in I_z$ and*

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

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

Assumption 2.9. *One of the following conditions holds:*

(i) for all $j = 1, \ldots, m$ we have $g_j(x) = g_j(x_c)$;

(ii) For every sequence of points $\{w_k\}$ such that $w_k \in X \cap \mathcal{Z}$, for all k and converging to the point $\bar{w} \in \mathcal{F} \cap \mathcal{Z}$, for all the sequences $\tilde{w}_k \in X \cap \mathcal{Z}$ such that for all k

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

there exist 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 2.7 is a standard assumption is constrained optimization and is mainly needed to guarantee existence of a solution. Assumption 2.8 also is quite standard and is needed to guarantee existence and boundedness of the lagrange multipliers. We note that Assumption 2.8 is well-posed thanks to the standing assumption that $I_c \neq \emptyset$. Finally, Assumption 2.9 is a more technical. Part (i) states that the constraints does not depend on the discrete variables. Part (ii) basically states that neighboring points of a feasible limit point $\bar{w}$ must not be on the boundary of the feasible set and is connected with forcing feasibility of points in the discrete neighborhood of the limit point.

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 towards 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, which is called SDFL, is convergent to strong stationary points. To achieve such a result, Algorithm SDFL performs a deeper investigation of the discrete neighborhoods by means of a local search procedure.

3. A line search algorithm model

In this section, we define a first derivative-free algorithm for MINLP, namely Algorithm DFL. The proposed method combines two basic ingredients, that 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 that defined in [9]. The presence of nonlinear constraints is accounted for by means of a derivative-free penalty approach like that described in [10]. The main parts of the method are the *Continuous search* and *Discrete search* procedures and the *Penalty parameter updating rule*. The Continuous search and Discrete search procedures are needed to explore the coordinate directions associated with, respectively, continuous and discrete variables. The current point is updated as soon as a sufficient reduction of the objective function is achieved by one of the procedures. The Continuous search procedure is quite standard in a derivative-free context and we refer the interested reader to [13]. The Discrete search procedure is similar to that defined in [9] and is governed by a control parameter ξ, which is reduced during the optimization process. In particular, the control parameter ξ is reduced whenever no discrete variable has been updated by the Discrete search procedure and the tentative steps along the discrete variables are equal to one. When no discrete variable has been updated during the iteration and the tentative steps for discrete variables are all equal to one, the method check if the penalty parameter has to be updated. All this considered, the convergence properties of Algorithm DFL can be characterized only with respect to a particular subsequence of iterates. The algorithm is described as follows.

Algorithm DFL

Data. $\theta \in (0,1)$, $q > 1$, $\epsilon_0 > 0$, $\xi_0 > 0$, $x_0 \in X \cap \mathcal{Z}$, $\tilde{\alpha}_0^i > 0$, $i \in I_c$, $\tilde{\alpha}_0^i = 1$, $i \in I_z$, set $d_0^i = e^i$, for $i = 1, \ldots, n$, and a sequence $\{\eta_k\} \downarrow 0$.

For $k = 0, 1, \ldots$
Set $y_k^1 = x_k$.
For $i = 1, \ldots, n$
If $i \in I_c$ **then** compute α by the *Continuous Search*$(\tilde{\alpha}_k^i, y_k^i, d_k^i, \epsilon_k; \alpha)$

If $\alpha = 0$ **then** set $\alpha_k^i = 0$ and $\tilde{\alpha}_{k+1}^i = \theta \tilde{\alpha}_k^i$.
else set $\alpha_k^i = \alpha$, $\tilde{\alpha}_{k+1}^i = \alpha$ and $d_{k+1}^i = d_k^i$.

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

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

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

End For
If $(y_k^{n+1})_z = (x_k)_z$ and $\tilde{\alpha}_k^i = 1$, $i \in I_z$, **then**
set $\xi_{k+1} = \theta \xi_k$, (*Penalty parameter updating rule*)
If $(\max_{i \in I_c}\{\alpha_k^i, \tilde{\alpha}_k^i\} \leq \epsilon_k^q)$ and $(\|g^+(x_k)\| > \eta_k)$, choose $\epsilon_{k+1} = \theta \epsilon_k$.
Else set $\epsilon_{k+1} = \epsilon_k$.
else set $\xi_{k+1} = \xi_k$.
Find $x_{k+1} \in X \cap \mathcal{Z}$ such that $P(x_{k+1}; \epsilon_k) \leq P(y_k^{n+1}; \epsilon_k)$.
End For

Then we report the Continuous search and Discrete search procedures (see [9]).

Continuous search $(\tilde{\alpha}, y, p, \epsilon; \alpha)$.

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

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

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

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

Step 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.

Step 5. Set $\alpha = 0$ and return.

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

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

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

Discrete search $(\tilde{\alpha}, y, p, \xi, \epsilon; \alpha)$**.**

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

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

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

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

Step 5. Set $\alpha = 0$ and return.

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

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

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

At every iteration k Algorithm DFL, starting from the current iterate x_k, explores all the coordinate directions and produces the intermediate points y_k^i, $i = 1, \ldots, n$. When $i \in I_c$, that is for the continuous variables, the actual steps α_k^i are computed and the tentative steps $\tilde{\alpha}_k^i$ are updated as described in [13]. When $i \in I_z$, the algorithm performs a Discrete search which is very similar to the Continuous search procedure except for the fact that the sufficient reduction is governed by the parameter ξ_k. The updating formula of tentative steps $\tilde{\alpha}_k^i$ is such that $1 \leq \tilde{\alpha}_k^i \in Z$. After having explored all the coordinate directions and if no discrete variable has been changed, the algorithm checks if the penalty parameter has to be updated.

Lemma 3.1. *Algorithm DFL is well-defined.*

Proof. In order to prove that Algorithm DFL is well defined, we have to ensure that both the Continuous search and Discrete search procedures, when performed along a direction d_k^i, with $i \in \{1, \ldots, n\}$, terminate in a finite number j of steps. This is clearly true since, by the instructions of the two procedures,

$$\begin{aligned} y_k^i + \delta^{-j}\alpha d_k^i \in X \qquad & \text{for all } i \in I_c, \\ y_k^i + 2^j \alpha d_k^i \in X \qquad & \text{for all } i \in I_z, \end{aligned}$$

and X, by Assumption 2.7, is a compact set. ∎

Lemma 3.2. *Let* $\{\xi_k\}$ *and* $\{\epsilon_k\}$ *be the sequences produced by Algorithm DFL. Then:*

(i)

$$\lim_{k \to \infty} \xi_k = 0; \tag{11}$$

(ii) the set

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

has infinitely many elements. Moreover, if $\lim_{k\to\infty} \epsilon_k = 0$*, then also*

$$K' = \{k : \xi_{k+1} < \xi_k, \epsilon_{k+1} < \epsilon_k\}$$

has infinitely many elements.

Proof. First we prove point (i). By the instructions of Algorithm DFL the sequence $\{\xi_k\}$ is monotonically non-increasing, that is, $0 < \xi_{k+1} \leq \xi_k$, for all k. Hence $\{\xi_k\}$ converges to a limit $M \geq 0$. Let us suppose, by contradiction, that $M > 0$. If this were the case, then an index $\bar{k} > 0$ would exist such that

$\xi_{k+1} = \xi_k = M$ and $\epsilon_{k+1} = \epsilon_k = \bar{\epsilon}$ for all $k \geq \bar{k}$. Moreover, for every index $k \geq \bar{k}$, a index $\bar{\imath} \in I_z$ (possibly depending on k) would exist such that

$$P(x_{k+1}; \bar{\epsilon}) \leq P(y_k^{\bar{\imath}} \pm \alpha_k^{\bar{\imath}} d_k^{\bar{\imath}}; \bar{\epsilon}) \leq P(y_k^{\bar{\imath}}; \bar{\epsilon}) - M \leq P(x_k; \bar{\epsilon}) - M, \tag{12}$$

otherwise the algorithm would have set $\xi_{k+1} = \theta \xi_k$. Relation (12) implies $P(x_k; \bar{\epsilon}) \to -\infty$ thus contradicting the assumption that $P(\cdot; \bar{\epsilon})$ is continuous on the compact set X, and this concludes the proof.

Finally, we prove point (ii). Point (i) and the updating rule of parameter ξ_k in Algorithm DFL imply that the set K 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' is infinite as well. ∎

Proposition 3.3. *Let $\{x_k\}$ be the sequence of points produced by Algorithm DFL and let K and K' be defined as in Lemma 3.2. Then,*

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

(ii) if $\lim_{k\to\infty} \epsilon_k = 0$, every limit point of $\{x_k\}_{K'}$ is stationary for Problem (1) with respect to the continuous variables.

Proof. Let us note that, by the instructions of Algorithm DFL, for all $k \in K$, $(y_k^{n+1})_z = (x_k)_z$ and $\tilde{\alpha}_k^i = 1$, $i \in I_z$. Hence, for all $k \in K$, the discrete variables are no longer updated.

We recall that, by the instructions of Algorithm DFL, at every iteration k, the following set of directions is considered

$$D_k = \{d_k^i, -d_k^i\}_{i \in I_c} = D^c.$$

At every iteration k, Algorithm DFL extracts information on the behavior of the penalty function along both d_k^i and $-d_k^i$.
In particular, along all d_k^i, $i \in I_c$, the algorithm identifies the following circumstances:
if the initial stepsize $\tilde{\alpha}_k^i$ fails to produce a decrease of the penalty function we have:

$$\text{either} \quad y_k^i + \tilde{\alpha}_k^i d_k^i \notin X, \tag{13}$$
$$\text{or} \quad P(y_k^i + \tilde{\alpha}_k^i d_k^i; \epsilon_k) > P(y_k^i; \epsilon_k) - \gamma(\tilde{\alpha}_k^i)^2; \tag{14}$$

if, instead, the initial stepsize $\tilde{\alpha}_k^i$ produces a decrease of the penalty function we have:

$$\text{both} \quad y_k^i + \tilde{\alpha}_k^i d_k^i \in X, \tag{15}$$
$$\text{and} \quad P(y_k^i + \tilde{\alpha}_k^i d_k^i; \epsilon_k) \leq P(y_k^i; \epsilon_k) - \gamma(\tilde{\alpha}_k^i)^2, \tag{16}$$

and a stepsize α_k^i is produced by the Continuous search such that:

$$\text{either} \quad y_k^i + \frac{\alpha_k^i}{\delta} d_k^i \notin X, \tag{17}$$
$$\text{or} \quad P(y_k^i + \frac{\alpha_k^i}{\delta} d_k^i; \epsilon_k) > P(y_k^i; \epsilon_k) - \gamma(\frac{\alpha_k^i}{\delta})^2; \tag{18}$$

As regards the behavior of the penalty function along the opposite direction $-d_k^i$, if (13) or (14) holds the algorithm investigates along the direction $-d_k^i$. Similarly to the analysis along d_k^i it determines that the initial stepsize $\tilde{\alpha}_k^i$ satisfies:

$$\text{either} \quad y_k^i + \tilde{\alpha}_k^i (-d_k^i) \notin X, \tag{19}$$
$$\text{or} \quad P(y_k^i + \tilde{\alpha}_k^i (-d_k^i); \epsilon_k) > P(y_k^i; \epsilon_k) - \gamma(\tilde{\alpha}_k^i)^2; \tag{20}$$

or it computes a stepsize α_k^i such that

$$\text{either} \quad y_k^i + \frac{\alpha_k^i}{\delta} (-d_k^i) \notin X, \tag{21}$$
$$\text{or} \quad P(y_k^i + \frac{\alpha_k^i}{\delta} (-d_k^i); \epsilon_k) > P(y_k^i; \epsilon_k) - \gamma(\frac{\alpha_k^i}{\delta})^2. \tag{22}$$

If (15) and (16) hold the algorithm does not consider the opposite direction $-d_k^i$ directly, but it can extract information on the behavior of P along $-d_k^i$ by using relation (16). In fact by setting $\tilde{y}_k^i = y_k^i + \tilde{\alpha}_k^i d_k^i$, relation (16) can be rewritten as:

$$P(\tilde{y}_k^i + \tilde{\alpha}_k^i(-d_k^i); \epsilon_k) \geq P(\tilde{y}_k^i; \epsilon_k) - \gamma(-(\tilde{\alpha}_k^i)^2). \tag{23}$$

Now let us consider the following (sub)sequence $\{x_k\}_{\bar{K}}$ where

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

$\bar{K} = K'$ if $\lim_{k\to\infty} \epsilon_k = 0$.

The instructions of Algorithm DFL imply that $x_k \in X$, for all k, so that, the sequence $\{x_k\}_{\bar{K}}$ admits limit points. Then, let $\bar{x} \in X$ be a limit point of $\{x_k\}_{\bar{K}}$. By using Lemma A.2 in Appendix we have:

$$\lim_{k\to\infty, k\in\bar{K}} \alpha_k^i = 0 \qquad \text{for } i \in I_c, \tag{24}$$

$$\lim_{k\to\infty, k\in\bar{K}} \tilde{\alpha}_k^i = 0 \qquad \text{for } i \in I_c. \tag{25}$$

By recalling the definitions of the search direction d_k^i, $i = 1, \ldots, n$ we obtain:

$$D \cap D(\bar{x}) \subseteq D_k. \tag{26}$$

Now by using (24), (25), (26) and Proposition 2.2 we have that, for sufficiently large k and for all $d_k^i \in D \cap D(\bar{x})$, neither (13) nor (17) can happen and that, for sufficiently large k and for all $-d_k^i \in D \cap D(\bar{x})$, neither (19) nor (21) can happen.

Now we prove the theorem by showing that all the requirements of Proposition A.1 in Appendix hold. Let us consider all the direction $d^i \in D \cap D(\bar{x})$.

If $d^i = d_k^i$, assumptions (59), (60) of Proposition A.1 in Appendix follow, for sufficiently large k, by setting $\eta_k^i = \tilde{\alpha}_k^i$, $y_k^i = y_k^i$ and $o(\eta_k^i) = \gamma(\tilde{\alpha}_k^i)^2$ if (14) holds or by setting $\eta_k^i = \frac{\alpha_k^i}{\delta}$, $y_k^i = y_k^i$ and $o(\eta_k^i) = \gamma(\frac{\alpha_k^i}{\delta})^2$ if (18) holds.

If $d^i = -d_k^i$, assumptions (59), (60) of Proposition A.1 in Appendix follow, for sufficiently large k, by setting $\eta_k^i = \tilde{\alpha}_k^i$, $y_k^i = y_k^i$ and $o(\eta_k^i) = \gamma(\tilde{\alpha}_k^i)^2$ if (20) holds, by setting $\eta_k^i = \frac{\alpha_k^i}{\delta}$, $y_k^i = y_k^i$ and $o(\eta_k^i) = \gamma(\frac{\alpha_k^i}{\delta})^2$ if (22) holds or by setting $\eta_k^i = \tilde{\alpha}_k^i$, $y_k^i = \tilde{y}_k^i$ and $o(\eta_k^i) = -\gamma(\tilde{\alpha}_k^i)^2$ if (23) holds.

Now, from the definitions of η_k^i, y_k^i and the fact that $y_k^i = x_k + \sum_{j=1}^{i-1} \alpha_k^j d_k^j$, we obtain

$$\frac{\max\limits_{i\in I_c}\{\eta_k^i, \|x_k - y_k^i\|\}}{\epsilon_k} \leq \frac{1}{\delta\epsilon_k} \max\left\{\max_{i\in I_c}\{\tilde{\alpha}_k^i, \alpha_k^i\}, \sum_{j=1}^{n} \alpha_k^j\right\}, \quad \forall\, i : d^i \in D \cap D(\bar{x}). \tag{27}$$

If $\bar{K} = K$ and $\lim_{k\to\infty} \epsilon_k = \bar{\epsilon} > 0$, point (61) of Proposition A.1 in Appendix follows from (24), (25) and (27).

If $\bar{K} = K'$ the instructions of Step 2 imply that, for all $k \in \bar{K}$,

$$\frac{\max\limits_{i\in I_c}\{\tilde{\alpha}_k^i, \alpha_k^i\}}{\epsilon_k} \leq \max_{i\in I_c}\{\tilde{\alpha}_k^i, \alpha_k^i\}^{\frac{p-1}{p}} \tag{28}$$

Then by (27) and (28) we obtain

$$\frac{\max_{i\in I_c}\{\eta_k^i, \|x_k - y_k^i\|\}}{\epsilon_k} \leq \frac{1}{\delta} \max\left\{\max_{i\in I_c}\{\tilde{\alpha}_k^i, \alpha_k^i\}^{\frac{p-1}{p}}, \sum_{j=1}^{n} \alpha_k^j\right\}, \quad \forall\, i : d^i \in D \cap D(\bar{x}), \tag{29}$$

which, along with (24) and (25), shows that (61) of Proposition A.1 in Appendix holds.

Finally, (62) of Proposition A.1 in Appendix follows from the updating rule of the penalty parameter ϵ_k of Algorithm DFL. ∎

Proposition 3.4. *Let $\{x_k\}$ be the sequence of points produced by Algorithm DFL. Let $K \subseteq \{1, 2, \ldots\}$ and $K' \subseteq K$ be defined as in Lemma 3.2. Then,*

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

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

Proof. Let us denote

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

Let $\bar{K} \subseteq \tilde{K}$ be an index set such that

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

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

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

that is, no discrete variable is updated by the Discrete search procedure.

Let us consider any point $\bar{x} \in \mathcal{N}_z(x^\star) \cap \mathcal{F}$. Then a direction $\bar{d} \in D(x^\star) \cap D^z$ exists such that

$$\bar{x} = x^\star + \bar{d}. \tag{30}$$

Recalling the definition of y_k^i in Algorithm DFL and the definition of the discrete neighborhood $\mathcal{N}_z(x)$, we have, for all $k \in \bar{K}$ and sufficiently large, that

$$(x^\star)_z = (x_k)_z = (y_k^i)_z, \quad i = 1, \ldots, n.$$

Further, by Lemma A.2 in Appendix, we have

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

Then, (30) implies

$$(x_k + \bar{d})_j = (y_k^i + \bar{d})_j = (x^\star + \bar{d})_j = (\bar{x})_j, \quad i = 1, \ldots, n, j \in I_z.$$

Now, Proposition 2.2 guarantees that for $k \in \bar{K}$ and sufficiently large, a direction $d_k^{\bar{i}} \in D(x_k) \cap D^z$ exists such that $d_k^{\bar{i}} = \bar{d}$, so that

$$(x_k + d_k^{\bar{i}})_j = (y_k^{\bar{i}} + d_k^{\bar{i}})_j = (x^\star + d_k^{\bar{i}})_j = (\bar{x})_j, \quad j \in I_z,$$

for all $k \in \bar{K}$ and sufficiently large. Hence, for k sufficiently large and $k \in \bar{K}$,

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

Then, we have

$$P(y_k^{\bar{i}} + d_k^{\bar{i}}; \epsilon_k) > P(y_k^{\bar{i}}; \epsilon_k) - \xi_k. \tag{31}$$

Recalling the expression of the penalty function $P(x; \epsilon)$ and the functions $\lambda_l(x; \epsilon)$ (defined in (10)), we can write

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

By using Proposition A.1 in Appendix and recalling that $x^\star \in \mathcal{F}$, we have

$$\lim_{k\to\infty, k\in\bar{K}} \lambda_l(y_k^{\bar{l}};\epsilon_k)\max\{0, g_l(y_k^{\bar{l}})\} = 0. \tag{32}$$

Therefore we obtain:

$$\lim_{k\to\infty, k\in\bar{K}} P(y_k^{\bar{l}};\epsilon_k) = f(x^\star). \tag{33}$$

Now if part (i) of Assumption 2.9 holds 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

$$\lim_{k\to\infty, k\in\bar{K}} P(y_k^{\bar{l}} + d_k^{\bar{l}};\epsilon_k) = f(\bar{x}). \tag{34}$$

If part (ii) of Assumption 2.9 holds, for sufficiently large $k \in \bar{K}$, 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

$$\lim_{k\to\infty, k\in\bar{K}} P(y_k^{\bar{l}} + d_k^{\bar{l}};\epsilon_k) = f(\bar{x}). \tag{35}$$

Finally, by making the limit in (31) and by using (34) and (35) we obtain

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

which completes the proof of the proposition. ∎

Theorem 3.5. *Let $\{x_k\}$ and $\{\epsilon_k\}$ be the sequences generated by Algorithm DFL. Let $K \subseteq \{1, 2, \ldots\}$ and $K' \subseteq K$ be defined as in Lemma 3.2. Then, $\{x_k\}$ admits limit points and*

(i) if $\lim_{k\to\infty} \epsilon_k = \bar{\epsilon}$, every limit point of $\{x_k\}_K$ is stationary for Problem (1);

(ii) if $\lim_{k\to\infty} \epsilon_k = 0$, every limit point of $\{x_k\}_{K'}$ is stationary for Problem (1).

Proof. By the instructions of Algorithm DFL, every iterate x_k belongs to X which, by Assumption 2.7, is compact. Hence $\{x_k\}$ admits limit points. Then, the proofs of Points (i)and (ii) follow by considering Propositions 3.3 and 3.4. ∎

4. An algorithm converging toward strong stationary points

In this subsection we propose another algorithm for the solution of Problem (1) and we prove that it is convergent to strong stationary points. In order to guarantee this stronger convergence property, a deeper investigation of the discrete neighborhood is carried out by a so-called "local search" procedure. The local search procedure first performs a line search along the direction related to a discrete variable. Then, if a point yielding a sufficient decrease of the penalty function is found, it becomes the current point. Otherwise, if a point z is found which is promising, that is, not significantly worse in penalty function value than the current point, a distributed search is performed starting from z.

Algorithm SDFL

Data. $\theta \in (0,1)$, $q > 1$, $\epsilon_0 > 0$, $\xi_0 > 0$, $x_0 \in X \cap \mathcal{Z}$, $\tilde{\alpha}_0^i > 0$, $i \in I_c$, $\tilde{\alpha}_0^i = 1$, $i \in I_z$, and set $d_0^i = e^i$, for $i = 1, \ldots, n$ and a sequence $\{\eta_k\} \downarrow 0$.

For $k = 0, 1, \ldots$
Set $y_k^1 = x_k$.
For $i = 1, \ldots, n$
If $i \in I_c$ **then** compute α by the *Continuous Search*$(\tilde{\alpha}_k^i, y_k^i, d_k^i, \epsilon_k; \alpha)$

If $\alpha = 0$ **then** set $\alpha_k^i = 0$ and $\tilde{\alpha}_{k+1}^i = \theta \tilde{\alpha}_k^i$.
else set $\alpha_k^i = \alpha$, $\tilde{\alpha}_{k+1}^i = \alpha$.
Set $d_{k+1}^i = d_k^i$.

else compute α by the *Local Search*$(\tilde{\alpha}_k^i, y_k^i, d_k^i, \xi_k, \epsilon_k; \alpha, \tilde{z})$

If $\alpha = 0$ and $\tilde{z} \neq y_k^i$ **then** $\alpha_k^i = 0$, $\tilde{\alpha}_{k+1}^i = \tilde{\alpha}_k^i$, set $y_k^{n+1} = \tilde{z}$, $d_{k+1}^i = d_k^i$ and **Exit For**
If $\alpha = 0$ **then**
compute α by the *Local Search*$(\tilde{\alpha}_k^i, y_k^i, -d_k^i, \xi_k, \epsilon_k; \alpha, \tilde{z})$
If $\alpha = 0$ and $\tilde{z} \neq y_k^i$ **then** set $\alpha_k^i = 0$, $\tilde{\alpha}_{k+1}^i = \tilde{\alpha}_k^i$,
$y_k^{n+1} = \tilde{z}$, $d_{k+1}^i = -d_k^i$ and **Exit For**
If $\alpha = 0$ **then** set $\alpha_k^i = 0$, $\tilde{\alpha}_{k+1}^i = \max\{1, \lfloor \tilde{\alpha}_k^i / 2 \rfloor\}$ and $d_{k+1}^i = d_k^i$.
else set $\alpha_k^i = \alpha$, $\tilde{\alpha}_{k+1}^i = \alpha$ and $d_{k+1}^i = -d_k^i$.
else set $\alpha_k^i = \alpha$, $\tilde{\alpha}_{k+1}^i = \alpha$ and $d_{k+1}^i = d_k^i$.

Endif

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

End For
If $(y_k^{n+1})_z = (x_k)_z$ and $\tilde{\alpha}_k^i = 1$, $i \in I_z$, **then**
set $\xi_{k+1} = \theta \xi_k$, (*Penalty parameter updating rule)*
If $(\max_{i \in I_c}\{\alpha_k^i, \tilde{\alpha}_k^i\} \leq \epsilon_k^q)$ and $(\|g^+(x_k)\| > \eta_k)$, choose $\epsilon_{k+1} = \theta \epsilon_k$
Else set $\epsilon_{k+1} = \epsilon_k$.
else set $\xi_{k+1} = \xi_k$.
Find $x_{k+1} \in X \cap \mathcal{Z}$ such that $P(x_{k+1}; \epsilon_k) \leq P(y_k^{n+1}; \epsilon_k)$.
End For

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

Data. $\nu > 0$.

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

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

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

Else go to Step 5.

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

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

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

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

Set $w^1 = z$.

For $i = 1, \ldots, n$

Let $q^i = e^i$.

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

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

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

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

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

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

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

End For

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

Algorithm SDFL along with the Local search procedure, generates some sequences and, in particular, the following ones: $\{x_k\}$, $\{\xi_k\}$, $\{y_k^i\}$, $\{d_k^i\}$, $\{\alpha_k^i\}$, $\{\tilde{\alpha}_k^i\}$, $\{z_k^i\}$, for $i = 1, \ldots, n$. Moreover, we remark that the Local search procedure can be viewed as a Discrete search enriched by a *Grid search* (see Step 5 of the Local search). More precisely, the Grid search is used to better explore the neighborhood of a promising point z with respect to the current point y, that is a point z such that $f(y) - \xi \leq f(z) < f(y) + \nu$.

Lemma 4.1. *The Local search procedure is well-defined.*

Proof. In order to prove that procedure Local search is well-defined, we need to show that the condition at Step 3 is eventually satisfied. Let us assume, by contradiction, the condition at Step 3 is never satisfied. If this was the case, then we would get a contradiction with the compactness of set X. □

Lemma 4.2. *Algorithm SDFL is well-defined.*

Proof. In order to prove that Algorithm SDFL is well defined, we have to ensure that, when performed along a direction d_k^i, with $i \in \{1, \ldots, n\}$, Step 1 and 2 of the Local search procedure are executed a

finite number j of times, since by Lemma 3.1 we already have that the Continuous search procedure is well-defined. By the instructions of the Local search procedure, when Step 2 is executed, we have

$$y_k^i + \delta^{-j}\alpha d_k^i \in X \qquad \text{for all } i \in I_c,$$

then, the proof follows by recalling that X, by assumption, is a compact set. ∎

Lemma 4.3. *Let $\{\xi_k\}$ and $\{\epsilon_k\}$ be the sequences produced by Algorithm SDFL. Then,*

(i)

$$\lim_{k\to\infty} \xi_k = 0; \tag{36}$$

(ii) the set

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

has infinitely many elements. Moreover, if $\lim_{k\to\infty} \epsilon_k = 0$, then also

$$K' = \{k : \xi_{k+1} < \xi_k, \epsilon_{k+1} < \epsilon_k\}$$

has infinitely many elements.

Proof. First we prove point (i). By the instructions of Algorithm SDFL the sequence $\{\xi_k\}$ is monotonically non-increasing, that is, $0 < \xi_{k+1} \leq \xi_k$, for all k. Hence $\{\xi_k\}$ converges to a limit $M \geq 0$. Let us suppose, by contradiction, that $M > 0$. If this was the case, then an index $\bar{k} > 0$ would exist such that $\xi_{k+1} = \xi_k = M$ and $\epsilon_{k+1} = \epsilon_k = \bar{\epsilon}$, for all $k \geq \bar{k}$. Moreover, for every index $k \geq \bar{k}$, and index $\bar{\imath} \in I_z$ (possibly depending on k) would exist such that

$$P(x_{k+1};\bar{\epsilon}) \leq P(y_k^{\bar{\imath}} \pm \alpha_k^{\bar{\imath}} d_k^{\bar{\imath}};\bar{\epsilon}) \leq P(y_k^{\bar{\imath}};\bar{\epsilon}) - M \leq P(x_k;\bar{\epsilon}) - M, \tag{37}$$

otherwise the algorithm would have set $\xi_{k+1} = \theta\xi_k$. Relation (37) implies $P(x_k;\bar{\epsilon}) \to -\infty$ thus contradicting the assumption that $P(\cdot;\bar{\epsilon})$ is continuous on the compact set X, and this concludes the proof.

Now, we prove point (ii). Point (i) and the updating rule of parameter ξ_k in Algorithm SDFL imply that the set K is infinite. Furthermore, if $\lim_{k\to\infty} \epsilon_k = 0$, the updating rule of Algorithm SDFL for ξ_k and ϵ_k implies that the set K' is infinite as well. ∎

Proposition 4.4. *Let $\{x_k\}$ and $\{\epsilon_k\}$ be the sequences produced by Algorithm SDFL. Let $K \subseteq \{1, 2, \ldots\}$ and $K' \subseteq K$ be defined as in Lemma 4.3. Then, $\{x_k\}$ admits limit points and*

(i) if $\lim_{k\to\infty} \epsilon_k = \bar{\epsilon}$, every limit point of $\{x_k\}_K$ is stationary for Problem (1);

(ii) if $\lim_{k\to\infty} \epsilon_k = 0$, every limit point of $\{x_k\}_{K'}$ 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 3.5. ∎

Proposition 4.5. *Let $\{x_k\}$ and $\{\epsilon_k\}$ be the sequences produced by Algorithm SDFL. Let $K \subseteq \{1, 2, \ldots\}$ and $K' \subseteq K$ be defined as in Lemma 4.3. Then, $\{x_k\}$ admits limit points and*

(i) if $\lim_{k\to\infty} \epsilon_k = \bar{\epsilon}$, every limit point of $\{x_k\}_K$ is strong stationary for Problem (1);

(ii) if $\lim_{k\to\infty} \epsilon_k = 0$, every limit point of $\{x_k\}_{K'}$ is strong stationary for Problem (1).

Proof. Let us denote

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

By the instructions of Algorithm SDFL, every iterate x_k belongs to X which, by Assumption 2.7 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 Strong Stationary Point, we have to show that a $\lambda^\star \in \Re^m$ exists such that the pair $(x^\star, \lambda^\star)$ satisfies (4), (5) and (6). Recalling the fact that the Local search is an enrichment of the Discrete search defined in subsection 3, the limit points produced by Algorithm SDFL surely satisfy (4), (5) and (6) which can be derived by using point (i) of Proposition 3.3 and Proposition 3.4.

Now we have to show that, for all $\bar{x} \in \mathcal{N}_z(x^\star) \cap \mathcal{F}$ such that $f(\bar{x}) = f(x^\star)$, it is possible to find a $\bar{\lambda} \in \Re^m$ such that the pair $(\bar{x}, \bar{\lambda})$ satisfies (7), (8) and (9). For any choice of $\bar{x} \in \mathcal{N}_z(x^\star) \cap \mathcal{F}$ such that $f(\bar{x}) = f(x^\star)$, and, reasoning as in Proposition 3.4, we can find a subsequence $\{z_k^{\bar{\imath}}\}_{\bar{K}}$, for some index $\bar{\imath} \in \{1, 2, \ldots, n\}$, such that,

$$\lim_{k\to\infty, k\in\bar{K}} z_k^{\bar{\imath}} = \bar{x}, \tag{38}$$

and, for all $k \in \bar{K}$ and sufficiently large, that

$$(\bar{x})_z = (z_k^{\bar{\imath}})_z.$$

Let us consider any point $\tilde{x} \in \mathcal{N}_z(\bar{x}) \cap \mathcal{F}$. Then a direction $\tilde{d} \in D(\bar{x}) \cap D^z$ exists such that

$$\tilde{x} = \bar{x} + \tilde{d}. \tag{39}$$

Then, (39) implies

$$(z_k^{\bar{\imath}} + \tilde{d})_j = (\bar{x} + \tilde{d})_j = (\tilde{x})_j, \quad j \in I_z.$$

Now, Proposition 2.2 guarantees that for $k \in \bar{K}$ and sufficiently large, a direction $d_k^{\bar{\jmath}} \in D(z_k) \cap D^z$ exists such that $d_k^{\bar{\jmath}} = \tilde{d}$, so that

$$(z_k^{\bar{\imath}} + d_k^{\bar{\jmath}})_j = (\bar{x} + d_k^{\bar{\jmath}})_j = (\tilde{x})_j, \quad j \in I_z,$$

for all $k \in \bar{K}$ and sufficiently large and

$$\lim_{k\to\infty, k\in\bar{K}} z_k^{\bar{\imath}} + d_k^{\bar{\jmath}} = \tilde{x}. \tag{40}$$

Hence, for k sufficiently large and $k \in \bar{K}$,

$$z_k^{\bar{\imath}} + d_k^{\bar{\jmath}} \in X \cap Z.$$

Then, we have

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

Recalling the expression of the penalty function P and the functions λ_l (defined in (10)), we can write

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

By using Proposition A.1 in Appendix and recalling that $x^\star \in \mathcal{F}$, we have

$$\lim_{k\to\infty, k\in\bar{K}} \lambda_l(y_k^{\bar{\imath}}; \epsilon_k)\max\{0, g_l(y_k^{\bar{\imath}})\} = 0. \tag{42}$$

Therefore we obtain:

$$\lim_{k\to\infty, k\in\bar{K}} P(y_k^{\bar{\imath}}; \epsilon_k) = f(x^\star). \tag{43}$$

Now we show that the following limits hold.

$$\lim_{k\to\infty, k\in\bar{K}} \lambda_l(z_k^{\bar{\imath}}; \epsilon_k)\max\{0, g_l(z_k^{\bar{\imath}})\} = 0, \tag{44}$$

$$\lim_{k\to\infty, k\in\bar{K}} \lambda_l(z_k^{\bar{\imath}} + d_k^{\bar{\jmath}}; \epsilon_k)\max\{0, g_l(z_k^{\bar{\imath}} + d_k^{\bar{\jmath}})\} = 0, \tag{45}$$

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

Now, if part (i) of Assumption 2.9 holds and considering that $z_k^{\bar{\imath}} = y_k^{\bar{\imath}} + d_k^{\bar{\imath}}$ and $\bar{\imath}, \bar{\jmath} \in I_z$, we have

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

Relations (44) and (45) follow from (42).

If, on the other and, part (ii) of Assumption 2.9 holds, by (38), (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

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

Finally, by (40) and (45), we can write

$$\lim_{k\to\infty, k\in\bar{K}} P(z_k^{\bar{\imath}} + d_k^{\bar{\jmath}}; \epsilon_k) = f(\tilde{x}). \tag{46}$$

Now, recalling (38) and that, by Lemma A.2 (in Appendix) and Lemma 4.3,

$$\lim_{k\to\infty, k\in\bar{K}} y_k^{\bar{\imath}} = x^\star, \qquad \lim_{k\to\infty, k\in\bar{K}} \xi_k = 0, \tag{47}$$

relation (9) follows by taking the limit for $k \to \infty, k \in \bar{K}$ in (41), and considering that, by assumption, $f(\bar{x}) = f(x^\star)$.

Then we show that point $\bar{x}$ is stationary with respect to the continuous variables.

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

$$P(z_k^{\bar{\imath}}; \epsilon_k) = P(w_k^1; \epsilon_k) \geq P(w_k^2; \epsilon_k) \geq \ldots \geq P(w_k^n; \epsilon_k) > P(y_k^{\bar{\imath}}; \epsilon_k) - \xi_k. \tag{48}$$

From (38) and (44) we obtain

$$\lim_{k\to\infty, k\in\bar{K}} P(z_k^{\bar{\imath}}; \epsilon_k) = f(\bar{x}). \tag{49}$$

Then, by (43), (47), (49) and by considering that, by assumption, $f(\bar{x}) = f(x^\star)$, we get

$$\lim_{k\to\infty, k\in\bar{K}} P(w_k^i; \epsilon_k) = f(\bar{x}), \qquad i = 1, \ldots, n. \tag{50}$$

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 A.2 in Appendix, $\tilde{\alpha}_k^i \to 0$, for all $i \in I_c$.

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

$$P(w_k^{i+1}; \epsilon_k) = P(w_k^i + \hat{\alpha}_k^i q_k^i; \epsilon_k) \leq P(w_k^i; \epsilon_k) - \gamma(\hat{\alpha}_k^i)^2, \tag{51}$$

we have that $\hat{\alpha}_k^i \to 0$, by (50) and (51). Hence, recalling that $w_k^1 = z_k^{\bar{\imath}}$ by definition of the Local search procedure, by (38), and the fact that $\tilde{\alpha}_k^i \to 0$ and $\hat{\alpha}_k^i \to 0$, we have that

$$\lim_{k\to\infty, k\in\bar{K}} w_k^i = \bar{x}. \tag{52}$$

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

$$\eta_k^i = \begin{cases} \tilde{\alpha}_k^i & \text{if} \quad P(w_k^i + \tilde{\alpha}_k^i \bar{d}^i; \epsilon_k) > P(w_k^i; \epsilon_k) - \gamma(\tilde{\alpha}_k^i)^2, \\ \dfrac{\hat{\alpha}_k^i}{\delta} & \text{if} \quad P\left(w_k^i + \dfrac{\hat{\alpha}_k^i}{\delta}\bar{d}^i; \epsilon_k\right) > P(w_k^i; \epsilon_k) - \gamma\left(\dfrac{\hat{\alpha}_k^i}{\delta}\right)^2. \end{cases} \tag{53}$$

Then, we can write

$$P(w_k^i + \eta_k^i \bar{d}^i; \epsilon_k) > P(w_k^i; \epsilon_k) - \gamma(\eta_k^i)^2. \tag{54}$$

By Lemma A.2 in Appendix, we have, for all $i \in I_c$, that

$$\lim_{k\to\infty, k\in\bar{K}} \eta_k^i = 0. \tag{55}$$

From (38) and (52) it follows that

$$\lim_{k\to\infty, k\in\bar{K}} \|z_k^{\bar{l}} - w_k^i\| = 0 \tag{56}$$

for all $i \in I_c$. Since, for $k \in \bar{K}$ and $k \geq \bar{k}$, $\epsilon_k = \bar{\epsilon}$ and by (38) and the fact that $\bar{x} \in \mathcal{F}$, we can write

$$\lim_{k\to\infty, k\in\bar{K}} \epsilon_k \|g^+(z_k^{\bar{l}})\| = 0. \tag{57}$$

Thus, considering that for k sufficiently large $w_k^i + \eta_k^i \bar{d}^i \in X \cap Z$, (54), (55), (56) and (57) prove that the hypotheses of Proposition A.1 in Appendix are satisfied. Hence, $\bar{x}$ is stationary with respect to the continuous variables. ∎

5. Numerical results

In this section we report the numerical performance of the proposed sequential penalty derivative-free algorithms DFL and $SDFL$ both on a set of academic test problems and on a real application arising in the optimal design of industrial motors. Moreover, a comparison with NOMAD, which is a well-known software package for derivative-free optimization [3], on the same set of test problems is carried out.

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: $\gamma = 10^{-6}$, $\theta = 0.5$, $p = 2$,

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

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

$$(\epsilon_0)^j = \begin{cases} 10^{-3} & \text{if } g_j(x_0)^+ < 1, \\ 10^{-1} & \text{otherwise.} \end{cases} \qquad j = 1, \ldots, m. \tag{58}$$

In order to preserve all the theoretical results, the test at Step 2 of Algorithm DFL $\max_{i=1,\ldots,n}\{\tilde{\alpha}_k^i, \alpha_k^i\} \leq \epsilon_k^p$ has been substituted by

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

As termination criterion, we stop the algorithm whenever $\max_{i\in I_c}\{\tilde{\alpha}_k^i, \alpha_k^i\} \leq 10^{-6}$. As a consequence of this stopping condition and of the initialization (58), we have that the final values of the penalty parameters are greater than 10^{-6}. Finally, we allow a maximum of 5000 function evaluations.

5.1. Results on test problems

We selected a set of 50 test problems from the well-known collections [7, 14] which have been suitably modified by letting some variables assume only a finite number of values. In particular, for every even index i, variable $x^i \in X^i$ with

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

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

PROBLEM	n	m	f_0	viol$_0$
HS 14	2	3	3.00E+00	1.0000000E+00
HS 15	2	2	3.00E+00	1.6090000E+03
HS 16	2	2	0.00E+00	5.8500000E+01
HS 18	2	2	0.00E+00	6.2504000E+02
HS 19	2	2	2.14E+03	2.8030301E+04
HS 20	2	3	1.25E+00	8.5000000E+00
HS 21	2	1	0.00E+00	-9.9960000E+01
HS 22	2	2	4.00E+00	1.0000000E+00
HS 23	2	5	3.00E+00	9.0000000E+00
HS 30	3	1	0.00E+00	2.0000000E+00
HS 31	3	1	0.00E+00	4.8250000E+01
HS 39	4	4	1.60E+01	-2.0000000E+00
HS 40	4	6	1.29E+00	0.0000000E+00
HS 42	4	4	2.00E+00	2.4000000E+01
HS 43	4	3	0.00E+00	0.0000000E+00
HS 60	3	2	9.76E+00	2.1000000E+01
HS 64	3	1	3.20E+06	7.2001940E+09
HS 65	3	1	0.00E+00	6.7400000E+01
HS 72	4	2	5.75E+00	2.0000300E+05
HS 74	4	8	1.40E+03	1.3440000E+03
HS 75	4	8	1.40E+03	1.3440000E+03
HS 78	5	6	8.00E+00	0.0000000E+00
HS 79	5	6	1.06E+01	4.1000000E+01
HS 80	5	6	8.00E+00	1.0000000E+00
HS 83	5	6	2.77E+00	-3.2217431E+04
HS 95	6	4	9.49E+01	1.0946900E+00
HS 96	6	4	1.75E+02	1.0946900E+00
HS 97	6	4	9.49E+01	1.0946900E+00
HS 98	6	4	2.85E+02	1.0946900E+00
HS 100	7	4	0.00E+00	1.1570000E+03
HS 101	7	6	3.70E+02	2.2063241E+03
HS 102	7	6	3.70E+02	2.2077641E+03
HS 103	7	6	3.70E+02	2.2105837E+03
HS 104	8	6	5.19E+00	6.7868285E-01
HS 106	8	6	1.35E+06	1.5500000E+04
HS 107	9	6	1.35E+00	2.9120000E+03
HS 108	9	13	3.00E+00	0.0000000E+00
HS 113	10	8	9.30E+01	1.3250000E+03
HS 114	10	14	1.18E+03	2.1914400E+03
HS 116	13	15	7.90E+02	2.2500005E+02
HS 223	2	2	0.00E+00	-1.0000000E-01
HS 225	2	5	3.00E+00	9.0000000E+00
HS 228	2	2	0.00E+00	0.0000000E+00
HS 230	2	2	1.00E+00	0.0000000E+00
HS 263	4	6	2.20E+03	-1.0000000E+01
HS 315	2	3	0.00E+00	0.0000000E+00
HS 323	2	2	1.00E+00	4.0000000E+00
HS 343	3	2	5.69E+02	-1.9416706E+01
HS 365∗	7	5	7.34E+00	6.0000000E+00
HS 369∗	8	6	5.45E+01	6.6000000E+03
HS 372∗	9	12	5.61E+02	3.5871700E+05
HS 373	9	12	1.01E+03	3.8151800E+05
HS 374	10	35	9.61E+00	0.0000000E+00

Table 1: Test problems characteristics

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 viol$_0 = \sum_{j=1}^{m} g_j(x_0)^+$. 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. In Table 5 we report the results obtained by using the three codes NOMAD, DFL and SDFL. In particular, we denote by:

- x_{bf} and x_{bi} the best feasible and infeasible solutions found by NOMAD;
- $x^\star$ the best solution found by the proposed algorithms DFL and SDFL;
- $h(x)$ the measure of the constraints violation;
- $f(x)$ the objective function value;
- nf the number of function evaluations.

We run NOMAD by using its default settings except for the MIN_MESH_SIZE which we set to 10^{-6} in order to be comparable with our stopping criterion.

As we can notice by taking a look at Table 5, NOMAD finds a feasible point (i.e. $h(x) \leq 10^{-6}$) in 24 problems, while DLF and SDFL find a feasible solution respectively in 34 and 35 problems. With respect to the number of function evaluations, it can be seen that DFL and SDFL perform better than NOMAD

on the average. Furthermore, SDFL is more costly than DFL but on seven problems (HS 65, HS 75, HS 102, HS 107, HS 114, HS 223, HS 343) is able to find points with a better objective function value than those found by DFL. We also report that on one problem (HS 83) SDFL converges to a point with a worse objective function value than that found by DFL.

Discrete variables				
		l.b.	u.b.	step
x1	Stack length [mm]	60	90	1
x2	Outer stator diameter [mm]	100	130	1
x14	Angle of flux barrier [deg.]	-10	10	1
x15	Angle of flux barrier [deg.]	-10	10	1
x16	Number of wires per slot	4	14	1
x17	Wire size [mm]	1.0	2.0	0.05

Table 2: Lower and upper bounds of discrete design variables

Continuous variables			
		l.b.	u.b.
x3	Inner stator diameter [mm]	72	80
x4	Stator tooth width [mm]	2.0	3.0
x5	Stator yoke thickness [mm]	3.0	5.0
x6	Slot opening width [mm]	1.2	1.6
x7	Slot opening depth [mm]	1.0	2.0
x8	Bottom loop radius [mm]	0.3	0.8
x9	Upper loop radius [mm]	0.3	0.8
x10	PM thickness [mm]	2.0	4.0
x11	Ratio of PM width to barrier width	0.80	0.95
x12	Magnet position [mm]	4.0	8.0
x13	Rotor tooth width [mm]	4.0	6.0

Table 3: Lower and upper bounds of continuous design variables

5.2. Results on an optimal design problem

In this section we report the results obtained by the three codes (NOMAD, DFL and SDFL) on a real optimal design problem. In particular, we consider the optimal design of Interior Permanent Magnet (IPM) synchronous motors [11] 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 high efficiency, high torque density, good overload capability 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 2 we also specify the allowed step. Figure 1 depicts a cross section of 1 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).

Nonlinear constraints		
	name	bound
g1	stator slot fill factor [%]	≤ 40
g2	max flux density in the stator tooth [T]	≤ 1.80
g3	max flux density in the stator yoke [T]	≤ 1.80
g4	linear current density [A/cm]	≤ 400
g5	efficiency at base speed [%]	≥ 90
g6	maximum speed [rpm]	≥ 20000
g7	back EMF at maximum speed [V]	≤ 120

Table 4: Bounds on the nonlinear constraints

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, they are black-box-type functions.

As concerns the solution of this real problem, DFL requires 915 function evaluations to locate a feasible point $x^\star$ with $f(x^\star) = 187.837$ whereas SDFL requires 3886 function evaluations to find a feasible point $x^\star$ with $f(x^\star) = 187.497$.

Finally, we ran NOMAD on the optimal design problem by using the same parameter settings as those used for test problems experimentation. NOMAD stopped for maximum number of black-box function evaluations (i.e. 20000 evaluations) reporting a feasible point $x^\star$ with $f(x^\star) = 187.982$. In order to better understand the behavior of NOMAD, it is worth noting that it finds a feasible solution $\bar{x}$ with $f(\bar{x}) = 194.124$ with 87 black-box function evaluations. Moreover, NOMAD finds $x^\star$ after 17701 function evaluations.

6. Conclusions

In this paper we have addressed MINLP problems. First, we have introduced the definition of stationary and strong stationary points and then we have proposed two algorithms and proved their convergence properties. The first algorithm, namely DFL, converges toward stationary points whereas the second algorithm, SDFL, converges toward strong 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 SDFL 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 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 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 SDFL is able to find better solutions than DFL at the cost of a higher number of function evaluations.

A. Technical results

In this section we report some technical results which are needed to prove convergence of DFL and SDFL. First we report the following general convergence result concerning continuous variables only.

Proposition A.1. *Let $\{\epsilon_k\}$ be a bounded sequence of positive penalty parameters. Let $\{x_k\}$ be a sequence of points such that $x_k \in X \cap Z$ for all k, and let $\bar{x} \in X \cap Z$ 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 $d^i \in D \cap D(\bar{x})$, $i \in I_c$, there exist vectors y_k^i and scalars $\eta_k^i > 0$ such that

$$y_k^i + \eta_k^i d^i \in X \cap \mathcal{Z}, \tag{59}$$

$$P(y_k^i + \eta_k^i d^i; \epsilon_k) \geq P(y_k^i; \epsilon_k) - o(\eta_k^i), \tag{60}$$

$$\lim_{k \to \infty, k \in K} \frac{\max_{i \in I_c : d^i \in D \cap D(\bar{x})} \{\eta_k^i, \|x_k - y_k^i\|\}}{\epsilon_k} = 0; \tag{61}$$

(ii)

$$\lim_{k \to \infty, k \in K} \epsilon_k \|g^+(x_k)\| = 0; \tag{62}$$

(iii)

$$(y_k^i)_z = (x_k)_z, \qquad \text{for all } i \in I_c. \tag{63}$$

Then $\bar{x}$ is a stationary point for Problem (1) with respect to the continuous variables, that is $\bar{x}$ satisfies (4) and (5) with $\bar{\lambda} \in \Re^m$ given by

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

where $\lambda_j(x; \epsilon)$, $j = 1, \ldots, m$, are defined in (10).

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 [10]. ∎

In the following lemma we show that the sequences of step sizes $\{\alpha_k^i\}$ and $\{\tilde{\alpha}_k^i\}$, $i \in I_c$, produced by DFL and SDFL with respect to the continuous admit a subsequence which converges to zero.

Lemma A.2. *Let $\{x_k\}$, $\{\xi_k\}$, $\{\epsilon_k\}$, $\{y_k^i\}$, $\{\alpha_k^i\}$, $\{\tilde{\alpha}_k^i\}$, $i \in I_c$ be the sequences produced by Algorithm DFL or SDFL. Then:*

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

$$\lim_{k \to \infty} \epsilon_k = \bar{\epsilon} > 0, \tag{64}$$

then, for all $i \in I_c$,

$$\lim_{k \to \infty} \alpha_k^i = 0, \tag{65}$$

$$\lim_{k \to \infty} \tilde{\alpha}_k^i = 0. \tag{66}$$

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

$$\lim_{k \to \infty} \epsilon_k = 0, \tag{67}$$

then, for all $i \in I_c$,

$$\lim_{k \to \infty, k \in H} \alpha_k^i = 0, \tag{68}$$

$$\lim_{k \to \infty, k \in H} \tilde{\alpha}_k^i = 0, \tag{69}$$

where $H = \{k : \epsilon_{k+1} < \epsilon_k\}$.

Proof. First we prove assertion (i). By (64), a $\bar{k} \geq 0$ exists such that

$$\epsilon_{k+1} = \epsilon_k = \epsilon_{\bar{k}} = \bar{\epsilon} \quad \text{for all } k \geq \bar{k}.$$

For every $i \in I_c$, we prove (65) by splitting the iteration sequence $\{k\}$ into two parts, $K^{'}$ and $K^{''}$. We identify with $K^{'}$ those iterations where

$$\alpha_k^i = 0 \tag{70}$$

and with $K^{''}$ those iterations where $\alpha_k^i \neq 0$ is produced by the Continuous search. Then the instructions of the algorithm imply

$$P(x_{k+1};\bar{\epsilon}) \leq P(y_k^i + \alpha_k^i d_k^i;\bar{\epsilon}) \leq P(y_k^i;\bar{\epsilon}) - \gamma(\alpha_k^i)^2 \|d_k^i\|^2 \leq P(x_k;\bar{\epsilon}) - \gamma(\alpha_k^i)^2 \|d_k^i\|^2. \tag{71}$$

Taking into account the compactness assumption on X, it follows from (71) that $\{P(x_k;\bar{\epsilon})\}$ tends to a limit $\bar{P}$. If $K^{''}$ is an infinite subset, recalling that $\|d_k^i\| = 1$ we obtain

$$\lim_{k\to\infty, k\in K^{''}} \alpha_k^i = 0. \tag{72}$$

Therefore, (70) and (72) imply (65).
In order to prove (66), for each $i \in I_c$ we split the iteration sequence $\{k\}$ into two parts, K_1 and K_2. We identify with K_1 those iterations where the Continuous search procedure, along the direction d_k^i, returns an $\alpha_k^i > 0$, for which we have:

$$\tilde{\alpha}_{k+1}^i = \alpha_k^i. \tag{73}$$

We denote by K_2 those iterations where we have failed in decreasing the objective function along the directions d_k^i and $-d_k^i$. By the instructions of the algorithm it follows that for all $k \in K_2$

$$\tilde{\alpha}_{k+1}^i \leq \theta \tilde{\alpha}_k^i, \tag{74}$$

where $\theta \in (0,1)$.
If K_1 is an infinite subset, from (73) and (65) we get that

$$\lim_{k\to\infty, k\in K_1} \tilde{\alpha}_{k+1}^i = 0. \tag{75}$$

Now, let us assume that K_2 is an infinite subset. For each $k \in K_2$, let m_k (we omit the dependence from i) be the biggest index such that $m_k < k$ and $m_k \in K_1$. Then we have:

$$\tilde{\alpha}_{k+1}^i \leq \theta^{(k+1-m_k)} \tilde{\alpha}_{m_k}^i \leq \tilde{\alpha}_{m_k}^i \tag{76}$$

(we can assume $m_k = 0$ if the index m_k does not exist, that is, K_1 is empty).
As $k \to \infty$ and $k \in K_2$, either $m_k \to \infty$ (namely, K_1 is an infinite subset) or $(k+1-m_k) \to \infty$ (namely, K_1 is finite). Hence, if K_2 is an infinite subset, (76) together with (75), or the fact that $\theta \in (0,1)$, yields

$$\lim_{k\to\infty, k\in K_2} \tilde{\alpha}_{k+1}^i = 0, \tag{77}$$

so that (66) is proved, and this concludes the proof of point (i).

Now we prove point (ii). If (67) holds, there must exist an infinite subset $K \subseteq \{0,1,\ldots\}$ such that $\epsilon_{k+1} < \epsilon_k$ for all $k \in K$. From the instructions of Algorithm DFL the penalty parameter is only updated when

$$\max_{i\in I_c}\{\alpha_k^i, \tilde{\alpha}_k^i\} \leq \epsilon_k^q. \tag{78}$$

Then the proof follows from (78) and (67). ∎

References

[1] M.A. Abramson, C. Audet, J.W. Chrissis and J.G. Walston, Mesh Adaptive Direct Search Algorithms for Mixed Variable Optimization. *Optimization Letters*, vol. 3(1): 35–47 (2009)

[2] M.A. Abramson, C. Audet and J.E. Dennis Jr., Filter Pattern Search Algorithms for Mixed Variable Constrained Optimization Problems. *Pacific Journal of Optimization*, vol. 3(3):477-500 (2007)

[3] C. Audet and J.E. Dennis Jr., Mesh adaptive direct search algorithms for constrained optimization. *SIAM Journal on Optimization*, vol. 17(1):188-217 (2006)

[4] C. Audet and J.E. Dennis Jr., A pattern search filter method for nonlinear programming without derivatives. *SIAM Journal on Optimization*, vol. 14(4):980-1010 (2004)

[5] C. Audet and J. E. Dennis Jr., Pattern search algorithms for mixed variable programming. *SIAM Journal on Optimization*, vol. 11(3): 573-594 (2001)

[6] C.-J. Lin, S. Lucidi, L. Palagi, A. Risi and M. Sciandrone, A decomposition algorithm model for singly linearly constrained problems subject to lower and upper bounds. *Journal of Optimization Theory and Applications*, vol. 141: 107–126 (2009)

[7] W. Hock and K. Schittkowski, *Test examples for nonlinear programming codes*, in Lecture notes in Economics and Mathematical Systems, n. 187, Springer-Verlag, Berlin, Heidelberg, New York, 1981.

[8] R.M. Lewis and V. Torczon, Pattern search algorithms for bound constrained minimization. *SIAM Journal on Optimization*, vol. 9(4):1082–1099 (1999)

[9] G. Liuzzi, S. Lucidi, F. Rinaldi, Derivative-free methods for bound constrained mixed-integer optimization. *Computational Optimization and Applications*, DOI 10.1007/s10589-011-9405-3 (2011)

[10] G. Liuzzi, S. Lucidi, M. Sciandrone, Sequential penalty derivative-free methods for nonlinear constrained optimization. *SIAM Journal on Optimization*, vol. 20:2814–2835 (2010)

[11] S. Lucidi, F. Parasiliti, F. Rinaldi, M. Villani, Finite Element Based Multi-Objective Design Optimization Procedure of Interior Permanent Magnet Synchronous Motors for Wide Constant-Power Region Operation. Submitted to *IEEE Transaction on Industrial Electronics* (2011)

[12] S. Lucidi, V. Piccialli, M. Sciandrone, An algorithm model for mixed variable programming. *SIAM Journal on Optimization*, vol. 14:1057–1084 (2005)

[13] S. Lucidi and M. Sciandrone, A Derivative-Free Algorithm for Bound Constrained Optimization. *Computationa Optimization and Applications*, vol. 21(2): 119–142 (2002)

[14] K. Schittkowski, *More test examples for nonlinear programming codes*, in Lecture notes in Economics and Mathematical Systems, n. 282, Springer-Verlag, Berlin, Heidelberg, New York, 1987.

[15] S. Lucidi, M. Sciandrone and P. Tseng, Objective-derivative-free methods for constrained optimization. *Mathematical Programming*, vol. 92(1):37–59 (2002)

[16] V. Torczon, On the convergence of pattern search algorithms. *SIAM Journal on Optimization*, vol. 7(1):1–25 (1997)

[17] L.N. Vicente, Implicitly and densely discrete black-box optimization problems. *Optimization Letters*, vol. 3:475-482 (2009)

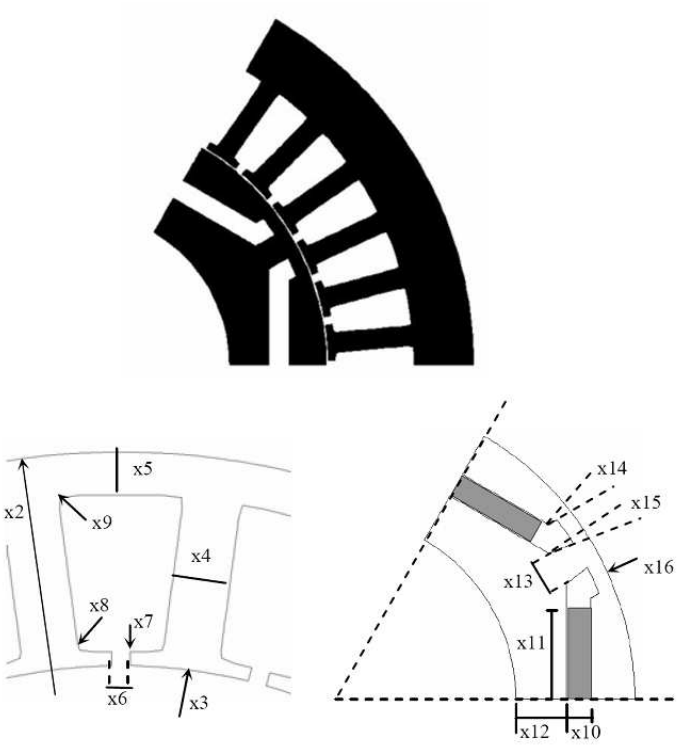

Figure 1: Cross section of the considered IPM motor (1 pole) and design variables, except for the wire size (x17).

	NOMAD				DFL			SDFL		
PROBLEM	$f(x_{bf})$	$f(x_{bi})$	$h(x_{bi})$	nf	$f(x^\star)$	$h(x^\star)$	nf	$f(x^\star)$	$h(x^\star)$	nf
HS 14	1.000000E+01	9.169918E+00	1.42E-01	59	1.000488E+00	2.50E-01	77	1.000488E+00	2.50E-01	77
HS 15	3.065000E+02	3.071846E+02	3.91E-03	54	3.065000E+02	0.00E+00	28	3.065000E+02	0.00E+00	28
HS 16	7.711097E-01	1.000391E+00	1.95E-04	90	5.850000E+01	0.00E+00	21	5.850000E+01	0.00E+00	21
HS 18	5.636114E+01	1.000625E+02	8.28E-03	387	1.562900E+02	0.00E+00	31	1.562900E+02	0.00E+00	31
HS 19	-	-9.520350E+02	6.53E-01	52	-7.951430E+03	8.93E-01	84	-7.951430E+03	8.93E-01	84
HS 20	5.850000E+01	1.585000E+02	7.50E-01	33	5.650000E+01	0.00E+00	30	5.650000E+01	0.00E+00	30
HS 21	-9.996000E+01	-	-	33	-9.996000E+01	0.00E+00	23	-9.996000E+01	0.00E+00	23
HS 22	1.000000E+00	9.745295E-01	2.88E-02	65	1.000000E+00	0.00E+00	50	1.000000E+00	0.00E+00	50
HS 23	3.001264E+01	2.994430E+01	5.57E-02	78	1.000000E+00	1.00E+00	61	1.000000E+00	1.00E+00	61
HS 30	1.000000E+00	-	-	122	1.000000E+00	0.00E+00	66	1.000000E+00	0.00E+00	83
HS 31	6.104042E+00	7.152371E+00	5.72E-06	3377	1.000000E+01	0.00E+00	88	1.000000E+01	0.00E+00	88
HS 39	-	-1.652565E+00	1.37E+00	257	-3.515625E-05	1.24E-09	157	-3.515625E-05	1.24E-09	157
HS 40	-	0.000000E+00	1.23E-03	180	0.000000E+00	0.00E+00	92	0.000000E+00	0.00E+00	92
HS 42	1.400000E+01	1.499908E+01	6.02E-04	681	1.400000E+01	0.00E+00	92	1.400000E+01	0.00E+00	100
HS 43	-4.400000E+01	-3.997727E+01	8.62E-05	1946	-2.577788E+01	0.00E+00	136	-2.577788E+01	0.00E+00	136
HS 60	-	1.526250E+02	7.36E-03	63	1.528679E+02	7.62E-07	147	1.528679E+02	7.62E-07	147
HS 64	2.299848E+04	2.351047E+04	7.65E-03	300	6.385192E+03	0.00E+00	422	6.385192E+03	0.00E+00	545
HS 65	9.588401E-01	1.379304E+00	1.11E-04	1659	5.224990E+00	0.00E+00	98	1.064343E+00	0.00E+00	654
HS 72	2.058468E+09	1.410032E+15	1.21E-04	5501	2.022679E+04	0.00E+00	430	2.022679E+04	0.00E+00	394
HS 74	-	4.864848E+03	7.70E+01	328	5.159206E+03	2.03E+01	620	5.159206E+03	2.03E+01	620
HS 75	-	5.219100E+03	7.84E+00	367	5.518396E+03	4.84E+01	5005	5.099690E+03	2.95E+01	4083
HS 78	0.000000E+00	0.000000E+00	1.33E-03	223	0.000000E+00	0.00E+00	131	0.000000E+00	0.00E+00	131
HS 79	-	9.789800E+01	1.24E+00	269	1.919152E+01	1.58E-02	218	1.919152E+01	1.58E-02	218
HS 80	-	1.000000E+00	8.52E-04	215	1.000000E+00	1.00E+00	290	1.000000E+00	1.00E+00	290
HS 83	-	-3.098811E+04	5.71E-01	396	-2.997681E+04	0.00E+00	265	-2.988572E+04	0.00E+00	566
HS 95	-	3.722298E+00	2.44E+01	478	3.722298E+00	2.44E+01	154	3.722298E+00	2.44E+01	181
HS 96	-	3.914600E+00	7.23E+01	411	3.914600E+00	9.72E+01	112	3.914600E+00	9.72E+01	116
HS 97	-	3.722298E+00	2.44E+01	478	3.722298E+00	2.44E+01	154	3.722298E+00	2.44E+01	181
HS 98	-	3.914600E+00	1.48E+02	416	3.914600E+00	2.07E+02	112	3.914600E+00	2.07E+02	116
HS 100	6.934424E+02	6.934418E+02	1.27E-05	14118	6.909346E+02	0.00E+00	261	6.909346E+02	0.00E+00	261
HS 101	2.612381E+03	2.765071E+03	1.33E-05	3960	2.677980E+03	0.00E+00	805	2.677980E+03	0.00E+00	745
HS 102	-	1.437486E+03	1.26E-01	1247	1.867129E+03	0.00E+00	465	1.849884E+03	0.00E+00	402
HS 103	1.371082E+03	1.465500E+03	4.41E-05	3140	1.233989E+03	0.00E+00	430	1.233989E+03	0.00E+00	410
HS 104	-	4.199728E+00	1.40E-02	636	4.200000E+00	1.25E-01	1228	4.200000E+00	6.59E-03	461
HS 106	-	4.417678E+03	7.49E-01	792	4.006764E+03	1.73E+00	443	4.003999E+03	1.75E+00	841
HS 107	5.679358E+03	5.679358E+03	3.50E-06	15828	9.333333E+03	2.28E-01	523	6.394794E+03	0.00E+00	1570
HS 108	-5.000000E-01	-4.998474E-01	1.57E-07	6558	-5.000000E-01	0.00E+00	218	-5.000000E-01	0.00E+00	218
HS 113	6.598558E+01	6.956341E+01	2.11E-04	20000	1.535013E+02	0.00E+00	450	1.535013E+02	0.00E+00	450
HS 114	-	-6.229771E+02	2.90E-01	1271	-6.545246E+02	1.85E-02	609	-6.793286E+02	6.09E-02	646
HS 116	-	1.025303E+02	5.41E-01	3873	5.000000E+01	1.25E-01	515	5.000000E+01	1.25E-01	1418
HS 223	-8.335205E-01	-8.424316E-01	1.96E-01	112	-4.758835E-01	0.00E+00	52	-8.340317E-01	0.00E+00	70
HS 225	6.002336E+00	5.985787E+00	1.42E-02	91	2.000000E+00	0.00E+00	49	2.000000E+00	0.00E+00	49
HS 228	-3.000000E+00	-2.999999E+00	9.54E-07	92	-3.000000E+00	0.00E+00	29	-3.000000E+00	0.00E+00	33
HS 230	1.000000E+00	1.000000E+00	1.84E-03	78	1.000000E+00	0.00E+00	64	1.000000E+00	0.00E+00	66
HS 263	0.000000E+00	-1.562357E-02	2.75E-04	151	1.414215E+00	7.01E-06	327	1.414215E+00	7.01E-06	333
HS 315	0.000000E+00	1.000000E+00	1.53E-07	43	0.000000E+00	0.00E+00	37	0.000000E+00	0.00E+00	37
HS 323	5.000000E+00	4.898660E+00	6.50E-04	311	5.000000E+00	0.00E+00	63	5.000000E+00	0.00E+00	63
HS 343	-5.684000E+00	-5.682464E+00	1.79E-06	2403	-2.861894E+00	0.00E+00	96	-5.646711E+00	0.00E+00	273
HS 365 *	-	2.826911E+01	9.16E-01	620	0.000000E+00	2.00E+00	1476	0.000000E+00	2.00E+00	1436
HS 369 *	2.100000E+03	2.554287E+03	1.48E-05	2229	2.100000E+03	0.00E+00	249	2.100000E+03	0.00E+00	283
HS 372 *	-	3.981150E+05	1.03E+02	20000	1.787733E+04	0.00E+00	918	1.782128E+04	0.00E+00	1128
HS 373	-	2.118780E+05	3.56E+02	20000	1.161109E+05	8.75E-02	840	1.161109E+05	8.75E-02	840
HS 374	1.000000E+00	1.000000E+00	1.55E-04	273	2.000000E+00	0.00E+00	335	2.000000E+00	0.00E+00	335

Table 5: Comparison between NOMAD, DFL and SDFL