A derivative-free approach to mixed integer constrained multiobjective nonsmooth optimization

G. Liuzzi∗ S. Lucidi†
April 15, 2022

Abstract
In this work, we consider multiobjective optimization problems with both bound con-
straints on the variables and general nonlinear constraints, where objective and constraint
function values can only be obtained by querying a black box. Furthermore, we consider the
case where a subset of the variables can only take integer values. We propose a new linesearch-
based solution method and show that it converges to a set of stationary points for the problem.
For what concerns the continuous variables, we employ a strategy for the estimation of the
Pareto frontier recently proposed in the literature and which takes advantage of dense sequences
of search directions. The subset of variables that must assume discrete values are dealt with
using primitive directions appropriately modified to take into account the presence of more
than one objective functions. Numerical results obtained with the proposed method on a set
of test problems and comparison with other solution methods show the viability and efficiency
of the proposed approach.

Keywords. Derivative-free multiobjective optimization, Mixed-integer problems, Lipschitz opti-
mization, Inequality constraints

AMS 90C30, 90C11, 90C56, 65K05, 49J52

1 Introduction
In many applications (especially coming from engineering or economic contexts) the simultaneous
minimization of several conflicting objective functions is to be considered within a feasible region
described by (possibly) nonlinear constraints. It is often the case that objective and/or constraint
function values are computed by using complex simulation programs so that first (or higher) order
information is not available. Furthermore, function values are typically affected by a certain degree
of numerical noise which prevents the use of approximation techniques to compute derivatives.
A further distinguishing feature of many multiobjective problems that arise in applications is the
presence of variables that can only assume integer or discrete values.

The aim of the paper is to develop and analyse a derivative-free method for the solution of mul-
tiobjective mixed integer constrained optimization problems. More specifically, we are interested
in the definition of an algorithm belonging to the class of “a posteriori” methods, namely those
algorithms that try to compute an approximation of the Pareto front for the problem. Indeed,
when multiple objective functions are present the definition of solution points brings along the
fact the many “non-dominated” (equivalent) solutions exist. Such solutions compose the so-called

∗ “Sapienza” Università di Roma, Dipartimento di Ingegneria Informatica Automatica e Gestionale “A. Ruberti”,
Via Ariosto 25, 00185 Rome, Italy and Consiglio Nazionale delle Ricerche, Istituto di Analisi dei Sistemi e Informatica
“A. Ruberti”, Via dei Taurini 19, 0005 Rome, Italy. liuzzi@diag.uniroma1.it
† “Sapienza” Università di Roma, Dipartimento di Ingegneria Informatica Automatica e Gestionale “A. Ruberti”,
Via Ariosto 25, 00185 Rome, Italy. lucidi@dis.uniroma1.it
Pareto front and there is great interest in trying to generate a set of points that approximate the Pareto front.

Hence, in this paper, we are interested in developing new globally convergent derivative-free methods with posterial articulation of preferences for nonlinearly constrained multiobjective minimization problems of the following form:

\[
\begin{align*}
\min \quad & F(x) = (f_1(x), \ldots, f_q(x))^	op \\
\text{s.t.} \quad & g_i(x) \leq 0, \ \text{for all } i = 1, \ldots, m \\
& l_i \leq x_i \leq u_i, \ \text{for all } i = 1, \ldots, n, \\
& x_i \in \mathbb{R} \ \text{for all } i \in I^c, \\
& x_i \in \mathbb{Z} \ \text{for all } i \in I^z,
\end{align*}
\]

where \( x \in \mathbb{R}^n, \ l, u \in \mathbb{R}^n, \) and \( I^c \cup I^z = \{1, 2, \ldots, n\}, \) with \( I^c \cap I^z = \emptyset \) and \( I^c, I^z \neq \emptyset. \) We assume \( l_i < u_i \) for all \( i = 1, \ldots, n, \) and \( l_i, u_i \in \mathbb{Z} \) for all \( i \in I^z. \) \( f_i : \mathbb{R}^n \to \mathbb{R}, i = 1, \ldots, q, g_i : \mathbb{R}^n \to \mathbb{R}, i = 1, \ldots, m. \) We denote by \( X \) and \( Z \) the following sets

\[
X = \{ x \in \mathbb{R}^n : l_i \leq x_i \leq u_i, \ i = 1, \ldots, n \}, \quad Z = \{ x \in \mathbb{R}^n : x_i \in \mathbb{Z}, i \in I^z \},
\]

and by \( \mathcal{F} \) the feasible set of problem (1), namely,

\[
\mathcal{F} = \{ x \in \mathbb{R}^n : g_i(x) \leq 0, \ i = 1, \ldots, m \} \cap X \cap Z.
\]

We note that, by definition, \( X \) is a compact set.

To solve problem (1) many methods have been proposed in the past years. These methods belong both to the class of deterministic and probabilistic methods.

Within the class of deterministic derivative-free methods for multiobjective mixed-integer problems, \texttt{BiMADS} manages bi-objective problems and is able to generate an approximation of the Pareto front by solving a series of single-objective formulations of the bi-objective problem. The single objective problems are solved by using the MADS algorithm for single objective optimization. \texttt{BiMADS} is able to manage also the presence of discrete or integer variables. The Directional Multi Search (DMS) algorithm proposed in \cite{11} (and successively studied and enhanced in \cite{12, 5, 10}) was the first method in which the iterate is defined by a set of points rather than a single point. DMS can handle multiobjective problems with more than two objective functions but it cannot be used when some of the variables must take integer or discrete values. In \cite{24} the DFMO algorithm has been proposed. DFMO is a linesearch-based derivative-free algorithm which evolves a set of non-dominated points that approximate Pareto stationary points in the limit. \cite{4} proposes \texttt{imfil}, an adaptation to multiobjective derivative-free problems of the well-known implicit filtering algorithm for single objective. The \texttt{imfil} method can be heuristically modified in such a way to produce a set of non-dominated points. More recently, in \cite{2} a new MADS-type algorithm has been proposed for multiobjective problems, namely \texttt{DMulti-MADS}. This method does not aggregate objectives and keeps a list of non dominated points which converges to a (local) Pareto set.

As concerns methods that try to approximate the set of global Pareto solutions, \texttt{MODIR} proposes and hybridization of the DIRECT-type algorithm \texttt{DIRMIN} for derivative-free global optimization \cite{23} with the aforementioned DFMO \cite{24}. In \cite{13} an algorithm is presented that alternates between initializing new searches, using a clever multistart strategy, and exploring promising subregions, by means of a directional direct search algorithm. Finally, in this context, the paper \cite{22} proposes a new branch-and-bound-based algorithm for smooth nonconvex multiobjective optimization problems with convex constraints. The algorithm computes an approximation of the set of Pareto global optimal solutions.

Probabilistic (and heuristic) approaches, like e.g. genetic algorithms \cite{14} and simulated annealing \cite{27}. 
Inspired by the ideas in [16] and [11], we describe a new exact-penalty-based linesearch approach (with sufficient decrease) for nonlinearly constrained MOO problems. In order to study the convergence properties of the proposed method, we carry out a preliminary theoretical analysis of the problem itself. We describe new optimality conditions that take explicitly into account the bound constraints and that are obtained by only assuming Lipschitz continuity of the problem functions. We also prove that the original problem is equivalent to a bound constrained problem obtained by penalizing the nonlinear constraints with an exact merit function. In particular, we introduce a merit function that penalizes the general nonlinear inequality constraints in each term of $F(x)$ and we resort to the minimization of a penalty function subject to the simple bound constraints. We would like to point out that some exact penalty methods for multiobjective optimization have already been proposed in the literature (see e.g. [17, 18] and references therein). Anyway, to the best of our knowledge, this is the first time that a penalty approach with explicit handling of the bound constraints for Lipschitz continuous multiobjective problems is studied. The proposed approach enables to handle also infeasible starting points. Furthermore, thanks to this exact penalty, we can adapt the derivative-free approach in [16] to the MOO case. This new approach gives us three relevant advantages:

- by means of the sufficient decrease we can avoid the use of integer lattices (see e.g. [11, 16]);

- the extrapolation phase allows us to better exploit a descent direction and hence to prove, under some density assumptions on the search directions, convergence to a set of Pareto stationary points (i.e. we prove that any accumulation point of the sequences generated by our method is a Pareto stationary point);

- thanks to the exact penalty approach, the starting point can be infeasible.

The fact that any accumulation point of the generated sequences is a Pareto stationary point, is an interesting theoretical result, since it is slightly stronger than the results reported in [11]. We would also like to note that, to the best of our knowledge, the idea of only penalizing the nonlinear constraints is new in the context of multiobjective derivative-free optimization.

In the last part of the paper, we test the linesearch approach on both bound constrained and nonlinearly constrained problems. The aim of the tests on bound constrained problems is understanding to what extent the theoretical properties of our method are helpful in practice. On the other hand, the goal of the tests on nonlinearly constrained problems is to show the effectiveness of the exact penalty approach when embedded in a DFO method for bound constrained multiobjective problems. For this reason we report the results obtained by both our algorithm and the globally convergent version of DMS.

1.1 Notations and preliminary material

We start the section by adapting the Lipschitz continuity to the mixed integer case.

**Definition 1.1 (Lipschitz continuity w.r.t. the continuous variables)** Let $h : \mathbb{R}^n \rightarrow \mathbb{R}$. We say that $h$ is Lipschitz continuous w.r.t. the continuous variables when $L_h$ exists such that, for all $x, y \in \mathbb{R}^n$ with $x_i = y_i$, for $i \in I^*$, it results

$$
\|h(x) - h(y)\| \leq L_h \|x - y\|.
$$

Then, we state the main assumption on the functions defining problem [1].
Assumption 1 (Lipschitz continuity) We assume that all the functions defining problem \([1]\), namely \(f_i, i = 1, \ldots, q\), and \(g_j, j = 1, \ldots, m\), are Lipschitz continuous w.r.t. the continuous variables (see Definition \([1,1]\)).

Definition 1.2 (Generalized directional derivative and generalized gradient) Let \(h : \mathbb{R}^n \to \mathbb{R}\) be a Lipschitz continuous function near \(x \in \mathbb{R}^n\) with respect to its continuous variables \(x_c\) (see Definition \([1,1]\)). The generalized directional derivative of \(h\) at \(x\) in the direction \(s \in \mathbb{R}^n\), with \(s_i = 0\) for \(i \in I^c\), is:

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

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

\[
\partial_c h(x) = \{v \in \mathbb{R}^n : v_i = 0, i \in I^c, \text{ and } h_c^{CI}(x; s) \geq s^T v \text{ for all } s \in \mathbb{R}^n, \text{ with } s_i = 0 \text{ for } i \in I^c\}.
\]

Given a vector \(v \in \mathbb{R}^n\), a subscript will be used to denote either one of its components \((v_i)\) or the fact that it is an element of an infinite sequence of vectors \((v_k)\). In case of possible misunderstanding or ambiguities, the \(i\)th component of a vector will be denoted by \((v)_i\).

Given two vectors \(u, v \in \mathbb{R}^n\), we use the following convention for vector equalities and inequalities:

\[
\begin{align*}
&u = v \iff u_i = v_i, \forall i = 1, \ldots, n, \\
&u < v \iff u_i < v_i, \forall i = 1, \ldots, n, \\
&u \leq v \iff u_i \leq v_i, \forall i = 1, \ldots, n, \text{ and } u \neq v, \\
&u \leq v \iff u_i \leq v_i, \forall i = 1, \ldots, n.
\end{align*}
\]

Note that \(u \geq v\) if and only if \(-u \leq -v\).

We denote by \(v^j\) the generic \(j\)th element of a finite set of vectors. Furthermore, vectors \(e_1, \ldots, e_n\) represent the coordinate unit vectors. Given two vectors \(a, b \in \mathbb{R}^n\), we denote by \(y = \max\{a, b\}\) the vector such that \(y_i = \max\{a_i, b_i\}, i = 1, \ldots, n\). Furthermore, given a vector \(v\) we denote by \(v^+ = \max\{0, v\}\). The projection of a point \(x\) onto the set \(X\) will be denoted by \([x]_{\|x\|}\). Finally, we denote the unit sphere in the origin by \(S(0, 1) = \{d \in \mathbb{R}^n : \|d\| = 1\}\), and \(\text{Co}(A)\) indicates the convex hull of the set \(A\). Given a point \(x\) and a scalar \(\rho > 0\), \(\mathcal{B}(x, \rho) = \{y \in \mathbb{R}^n : \|x - y\| \leq \rho\}\).

We denote:

\[
\Gamma = \{\mu \in \mathbb{R}^q : \mu \geq 0, \sum_{i=1}^q \mu_i = 1\}.
\]

Finally, by \(1\) we denote the vector of all ones of dimension \(q\).

When dealing with several objective functions at a time, the concept of Pareto dominance is usually considered in the comparison of two points.

Definition 1.3 (Pareto dominance) Given two points, \(x, y \in \mathcal{F}\), we say that \(x\) dominates \(y\) if \(F(x) \leq F(y)\).

Anyway, when coming to optimality, it may not be possible to find a point which is optimal for all the objectives simultaneously. This is the reason why the concept of Pareto dominance is also used to characterize global and local optimality into a multiobjective framework. More specifically, by means of the following two definitions, we are able to identify a set of nondominated points (the so-called Pareto front or frontier) which represents the (global or local) optimal solutions of a given multiobjective problem.

Definition 1.4 (Global Pareto optimality) A point \(x^* \in \mathcal{F}\) is a global Pareto optimizer of Problem \([1]\) if it does not exist a point \(y \in \mathcal{F}\) such that \(F(y) \leq F(x^*)\).
Then, we have to define local Pareto solutions. Hence, we have to formally state the definition of neighborhoods in the mixed integer contexts. To this end, with respect to the discrete variables, we have to preliminary introduce the concept of feasible primitive directions which will be used to define discrete neighborhoods.

**Definition 1.5 (set of feasible primitive directions)** Given \( x \in X \cap Z \),

\[
D^z(x) = \{ d \in \mathbb{Z}^n : \text{d is primitive} \}, \quad d_c = 0, \quad x + d \in X \cap Z \}.
\]

It can be proved that \( \bar{D} = \bigcup_{x \in X \cap Z} D^z(x) \) is finite.

**Definition 1.6 (Neighborhoods)** Given \( \bar{x} \in X \cap Z \) and \( \rho > 0 \),

\[
B^z(\bar{x}) = \{ x \in X \cap Z : x = \bar{x} + d, \quad \forall d \in D^z(\bar{x}) \},
\]

\[
B^c(\bar{x}; \rho) = \{ x \in X : x_z = \bar{x}_z, \quad \| x_c - \bar{x}_c \| \leq \rho \}.
\]

Finally, we can give the definition of local Pareto solutions for problem (1).

**Definition 1.7 (local Pareto optimum)** \( x^* \in X \cap Z \) is a local Pareto optimum of problem (1) if

1. there is no \( y \in B^z(x^*) \cap F \) s.t. \( F(y) \leq F(x^*) \)
2. there is no \( y \in B^c(x^*, \rho) \cap F \) s.t. \( F(y) \leq F(x^*) \) for some \( \rho > 0 \).

As usual in the continuous optimization setting, we have now to introduce the definition of “stationary” points. For this reason, and to explicitly handle the simple bound constraints, we first need to formally define the cone of feasible directions for the bound constraints and w.r.t. the continuous variables.

**Definition 1.8 (Set of feasible continuous directions)** Given a point \( x \in X \cap Z \), the set

\[
D^c(x) = \{ s \in \mathbb{R}^n : s_i = 0 \quad \text{for all} \quad i \in I_z, \neq, s_i \geq 0 \quad \text{for all} \quad i \in I_c \quad \text{and} \quad x_i = l_i, \\
\quad s_i \leq 0 \quad \text{for all} \quad i \in I_c \quad \text{and} \quad x_i = u_i, \\
\quad s_i \in \mathbb{R} \quad \text{for all} \quad i \in I_c \quad \text{and} \quad l_i < x_i < u_i, \}
\]

is the cone of feasible continuous directions at \( x \) with respect to \( X \cap Z \).

Finally, stationary points of problem (1) can be defined.

**Definition 1.9 (Pareto-Clarke Stationarity w.r.t. continuous variables)** A point \( x^* \in F \) is a Pareto-Clarke stationary point w.r.t. the continuous variables when there exist multipliers \( \sigma_1^*, \ldots, \sigma_q^*, \lambda_1^*, \ldots, \lambda_m^* \in \mathbb{R} \), such that

\[
\sigma_i^* \geq 0, \quad \sigma^* \neq 0, \quad i = 1, \ldots, q, \\
\lambda_j^* \geq 0, \quad \lambda_j^* g_j(x^*) = 0, \quad j = 1, \ldots, m,
\]

and a vector

\[
\bar{\xi} \in \sum_{i=1}^q \sigma_i^* \partial f_i(x^*) + \sum_{j=1}^m \lambda_j^* \partial g_j(x^*),
\]

such that

\[
\bar{\xi}^\top d \geq 0, \quad \text{for every} \quad d \in D^c(x^*).
\]

Then, we say that a point \( x^* \in F \) is (Pareto) stationary for problem (1) when:

(i) it is Pareto-Clarke stationary w.r.t. the continuous variables and
(ii) it is a local Pareto optimum w.r.t. the discrete variables.

**Proposition 1.10 (Pareto-Clarke KKT Necessary Optimality Conditions)** Let \( x^* \in \mathcal{F} \) be a local Pareto minimum of the problem \([1]\) and assume that a direction \( d \in D^c(x^*) \) exists such that for all \( j \in \{1, \ldots, m : g_j(x^*) = 0\} : 

\[
(\xi^{g_j})^\top d < 0, \quad \text{for all } \xi^{g_j} \in \partial c g_j(x^*). \tag{3}
\]

Then, \( x^* \) is a Pareto stationary point.

**Proof.** The proof easily follows recalling definition 1.7 and the proof of [24, Proposition 3.4]. \( \square \)

### 2 The bound constrained case

\[
\min_{x \in X \cap Z} F(x) = (f_1(x), \ldots, f_q(x))^\top
\]

**Proposition 2.1** A point \( \bar{x} \in X \cap Z \) is a Pareto-Clarke stationary point of \([4]\) w.r.t. the continuous variables, if and only if for all \( d \in D^c(\bar{x}) \), an index \( j_d \in \{1, \ldots, q\} \) exists such that:

\[
(f_{j_d})^{Cl}(\bar{x}; d) \geq 0.
\]

**Proof.** The proof is an almost trivial repetition of [24, Proposition 3.7]. \( \square \)

We remark that a point \( \bar{x} \in X \cap Z \) is (Pareto) stationary for problem \([4]\) if and only if:

(i) it is Pareto-Clarke stationary w.r.t. the continuous variables and

(ii) it is a local Pareto optimum w.r.t. the discrete variables.

Given a point \( x \in X \cap Z \), the Clarke-Jahn generalized directional derivative of a function \( f \) along the direction \( d \in D^c(x) \) is given by (see [19, Section 3.5]):

\[
f^c(x; d) = \limsup_{\substack{y \to x, y \in X \cap Z \\cap \ \ y = x, y \in X \cap Z \\ cap \ \ t \downarrow 0, y + td \in X \cap Z}} \frac{f(y + td) - f(y)}{t}. \tag{5}
\]

Recall that, according to definitions [2] and [5], we have, for any \( d \in \mathbb{R}^n \),

\[
f^{Cl}(x; d) \geq f^c(x; d). \tag{6}
\]

**Definition 2.2** A point \( x^* \in X \cap Z \) is a Pareto-Clarke-Jahn stationary point w.r.t. the continuous variables if, for all \( d \in D^c(x^*) \), an index \( j_d \in \{1, \ldots, q\} \) exists such that

\[
(f_{j_d})^{Cl}(\bar{x}; d) \geq 0.
\]

**Lemma 2.3** With respect to the continuous variables, every Pareto-Clarke-Jahn stationary point \( x^* \in X \cap Z \) is also a Pareto-Clarke stationary point.

**Proof.** Recalling the definitions [2] and [5] of Clarke and Clarke-Jahn generalized directional derivatives, respectively, and relation [6], we have that

\[
(f_{j_d})^{Cl}(\bar{x}; d) \geq (f_{j_d})^c(\bar{x}; d).
\]

Hence, the result easily follows. \( \square \)
3 The penalty approach

Given problem (1), we introduce the following nonsmooth penalty functions

\[ Z_i(x; \epsilon) = f_i(x) + \frac{1}{\epsilon} \sum_{j=1}^{m} \max \{0, g_j(x)\} \]

and consider the bound constrained problem

\[
\begin{align*}
\min_{x \in X \cap Z} & \quad Z(x; \epsilon) = (Z_1(x; \epsilon), \ldots, Z_q(x; \epsilon))^T \\
\text{s.t.} & \quad x \in X \cap Z
\end{align*}
\]

(7)

Assumption 2 (EMFCQ for mixed-integer problems) Given Problem (1), for any \( x \in (X \cap Z) \setminus \hat{\mathcal{F}} \), one of the following conditions holds:

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

\[(\xi^g)^\top s \leq 0,\]

for all \( \xi^g \in \partial_c g_i(x) \) with \( i \in \{1, \ldots, m\} : g_h(x) \geq 0 \);

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

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

We first prove the equivalence between local minimum points and global minimum points of the original problem (1) and the penalized problem (7).

Proposition 3.1 Let Assumption 2 hold. A threshold value \( \epsilon^* > 0 \) exists such that the function \( P(x; \epsilon) \) has no Pareto-Clarke stationary points in \( (X \cap Z) \setminus \mathcal{F} \) for any \( \epsilon \in (0, \epsilon^*). \)

Proof. We proceed by contradiction and assume that for any integer \( k \), an \( \epsilon_k \leq 1/k \) and a stationary point for Problem (7) \( x_k \in (X \cap Z) \setminus \mathcal{F} \) exist. Then, let us consider a limit point \( \bar{x} \in (X \cap Z) \setminus \hat{\mathcal{F}} \) of the sequence \( \{x_k\} \) and, without loss of generality, let us call the corresponding subsequence as \( \{x_k\} \) as well. Then, since \( x_k \to \bar{x} \), the discrete variables remain fixed, i.e., \( (x_k)_i = \bar{x}_i \), for all \( i \in I^c \) and \( k \) sufficiently large. Now, the proof continues by separately assuming that point (i) or (ii) of Assumption 2 holds.

First we assume that point (i) of Assumption 2 holds at \( \bar{x} \). Therefore, a direction \( \bar{s} \in D^c(\bar{x}) \) exists such that

\[
(\xi^g)^\top \bar{s} < 0 \quad \text{for all} \quad \xi^g \in \partial_c g_i(\bar{x}) \quad \text{with} \quad i \in \{1, \ldots, m\} : g_h(\bar{x}) \geq 0.
\]

In particular, it follows that

\[
\max_{\xi^g \in \partial_c g_i(\bar{x})} (\xi^g)^\top \bar{s} = -\eta < 0,
\]

where \( I(\bar{x}) = \{i \in \{1, \ldots, m\} : g_i(\bar{x}) = \bar{\phi}(\bar{x})\} \), \( \bar{\phi}(\bar{x}) = \max \{0, g_1(x), \ldots, g_m(x)\} \) and \( \eta \) is a positive scalar. Note that \( \bar{x} \not\in \mathcal{F} \) implies \( \bar{\phi}(\bar{x}) > 0 \).

By Proposition 2.3, it follows that \( \bar{s} \in D^c(x_k) \). Moreover, since \( x_k \) is a stationary point, we have that an index \( j \in \{1, \ldots, q\} \) exists such that:

\[
\max_{\xi \in \partial_c Z_j(x_k; \epsilon)} (P_j)_{\epsilon}^C (x_k; \epsilon, \bar{s}) \geq 0.
\]
and there is no \( y \in B^2(x_k) \) s.t. \( F(y) \leq F(x_k) \).

By [8], we know that

\[ \partial_c Z_j(x_k; \bar{c}) \subseteq \partial_c f_j(x_k) + \frac{1}{\varepsilon} \partial_c(\max \{0, g_1(x_k), \ldots, g_m(x_k)\}) \tag{10} \]

and

\[ \partial(\max \{0, g_1(x_k), \ldots, g_m(x_k)\}) \subseteq \text{Co}\left(\{\partial_c g_i(x_k) : i \in I(x_k)\}\right), \tag{11} \]

where \( \text{Co}(A) \) denotes the convex hull of a set \( A \) (see [26, Theorem 3.3]).

Therefore, by (9)–(11), \( \xi^i_k \in \partial_c f_j(x_k) \), \( \xi^{g_i}_k \in \partial_c g_i(x_k) \) and \( \beta^i_k \) with \( i \in I(x_k) \) exist such that

\[ \left( \xi^i_k + \frac{1}{\varepsilon_k} \sum_{i \in I(x_k)} \beta^i_k \xi^{g_i}_k \right)^\top \bar{s} \geq 0, \tag{12} \]

\[ \sum_{i \in I(x_k)} \beta^i_k = 1 \text{ and } \beta^i_k \geq 0. \]

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

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

\[ \xi^i_k \rightarrow \xi^i, \tag{13a} \]

\[ \xi^{g_i}_k \rightarrow \xi^{g_i} \text{ for all } i \in \bar{I}, \tag{13b} \]

\[ \beta^i_k \rightarrow \beta^i \text{ for all } i \in \bar{I}. \tag{13c} \]

Now the upper semicontinuity of \( \partial_c f_j \) and \( \partial_c g_i \), with \( i \in \bar{I} \), at \( \bar{x} \) (see Proposition 2.1.5 in [17]) implies that \( \xi^i \in \partial_c f_j(\bar{x}) \) and \( \xi^{g_i} \in \partial_c g_i(\bar{x}) \) for all \( i \in \bar{I} \).

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

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

and, in turn, this implies that for \( k \) sufficiently large

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

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

\[ \bar{I} \subseteq I(\bar{x}). \tag{14} \]

Finally, for \( k \) sufficiently large, [8], (13), and (14) imply

\[ (\xi^{g_i}_k)^\top \bar{s} \leq -\frac{\eta}{2} \text{ for all } i \in \bar{I}, \tag{15} \]

and (12), multiplied by \( \varepsilon_k \), implies

\[ \left( \varepsilon_k \xi^i_k + \sum_{i \in \bar{I}} \beta^i_k \xi^{g_i}_k \right)^\top \bar{s} \geq 0. \tag{16} \]

Equations (15) and (16) yield

\[ 0 \leq \left( \varepsilon_k \xi^i_k + \sum_{i \in \bar{I}} \beta^i_k \xi^{g_i}_k \right)^\top \bar{s} \leq \left( \varepsilon_k \xi^i_k \right)^\top \bar{s} - \frac{\eta}{2}. \]
which, by using (13), gives rise to a contradiction when \( \varepsilon_k \to 0 \).

Now we assume that point \((ii)\) of Assumption \(2\) holds at \( \bar{x} \). Let \( \bar{d} \in D^z(\bar{x}) \) be the direction such that

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

(17)

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

\[
Z(x_k; \varepsilon_k) \leq Z(x_k + \bar{d}; \varepsilon_k),
\]

or an index \( j \) exists such that

\[
Z_j(x_k; \varepsilon_k) < Z_j(x_k + \bar{d}; \varepsilon_k),
\]

where \( x_k + \bar{d} \in B^z(x_k) \).

Hence, an index \( h \) exists such that

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

Multiplying by \( \varepsilon \) and considering that \( \varepsilon_k \to 0 \), if we take the limit for \( k \to \infty \) we have that

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

The latter equation is in contradiction with (17).

Now, we can prove that there exists a threshold value \( \bar{\varepsilon} \) for the penalty parameter such that, for any \( \varepsilon \in (0, \bar{\varepsilon}) \), any local minimum of the penalized problem is also a local minimum of the original problem.

**Proposition 3.2** For any \( \varepsilon > 0 \), every Pareto-Clarke stationary point \( \bar{x} \) of Problem \( 7 \), such that \( \bar{x} \in F \), is also a Pareto-Clarke stationary point of Problem \( 4 \).

**Proof.** Since \( \bar{x} \) is, by assumption, a Pareto-Clarke stationary point of Problem \( 7 \), then by Definition \( 9 \) we know that a vector of non-negative multipliers \( \mu \in \Gamma \), i.e., \( \mu \geq 0 \), \( \sum_{i=1}^{q} \mu_i \), not all zero, and a vector \( \xi^* \in \sum_{i=1}^{q} \mu_i \partial_c Z_i(\bar{x}, \varepsilon) \) exist such that, for all \( d \in D^z(\bar{x}) \),

\[
(\xi^*)^T d \geq 0.
\]

Now, we recall that

\[
\partial_c Z_i(x, \varepsilon) \subseteq \partial_c f_i(x) + \frac{1}{\varepsilon} \sum_{j \in I(x)} \partial_c g_j(x),
\]

where \( I(x) = \{ i : g_i(x) \geq 0 \} \). Hence, we have that \( \xi^* \in \sum_{i=1}^{q} \mu_i \left( \partial_c f_i(\bar{x}) + \frac{1}{\varepsilon} \sum_{j \in I(\bar{x})} \partial_c g_j(\bar{x}) \right) \), \( \mu \in \Gamma \). Then, denoting \( \lambda_j = (\sum_{i=1}^{q} \mu_i) / \varepsilon \),

\[
\max \left\{ \xi^T d : \xi \in \sum_{i=1}^{q} \mu_i \partial_c f_i(\bar{x}) + \sum_{j \in I(\bar{x})} \lambda_j \partial_c g_j(\bar{x}) \right\} \geq 0
\]
for all $d \in D^c(\bar{x})$ with $\lambda_j \geq 0$, $j \in I(\bar{x})$. The above condition shows that a $\bar{\xi} \in \sum_{i=1}^{q} \mu_i \partial_c f_i(\bar{x}) + \sum_{j \in I(\bar{x})} \lambda_j \partial_d g_j(\bar{x})$ exists such that $\bar{\xi}^\top d \geq 0$, for all $d \in D^c(\bar{x})$. Hence, by recalling that $\bar{x} \in \mathcal{F}$ and $I(\bar{x}) = I_0(\bar{x})$ when $\bar{x} \in \mathcal{F}$, and setting $\lambda_j = 0$ when $j \notin I_0(\bar{x})$, it is proved that $\bar{x}$ is a Pareto-Clarke stationary point of Problem (1).

Now, we introduce an intermediate result which basically states that, for $\epsilon$ sufficiently small, every Pareto-Clarke-Jahn stationary point of Problem (7) is a Pareto-Clarke stationary point of Problem (1).

**Proposition 3.3** Let Assumption 2 hold. Given problem (1) and considering problem (7), a threshold value $\epsilon^* > 0$ exists such that, for every $\epsilon \in (0, \epsilon^*)$, every Pareto-Clarke-Jahn-stationary point of Problem (7) is a Pareto-Clarke stationary point of Problem (1).

**Proof.** Let $\bar{x} \in X \cap Z$ be Pareto-Clarke-Jahn stationary for Problem (7). By (6), we also have that $\bar{x}$ is Pareto-Clarke stationary for Problem (7). Now, the proof follows by considering Propositions 3.1 and 3.2.

We conclude by reporting from [24] a result concerning correspondence between global Pareto minimizers of problems (7) and (1).

**Proposition 3.4** ([24, Proposition 3.13]) Let Assumption 2 hold. Then, given Problem (1) and considering Problem (7), a threshold value $\epsilon^* > 0$ exists such that, for every $\epsilon \in (0, \epsilon^*)$, any global Pareto minimizer of Problem (7) is a global Pareto minimizer of Problem (1), and conversely.

4 The algorithm

In the definition of Algorithm DFMOINT, and in particular in Phase 2.a (Explore discrete variables), we shall make use of the following notation. Given a list

$$L = \{(x_j, \alpha_j^c, \alpha_j^{(d)}, \xi_j) : j = 1, 2, \ldots, |L|\}$$

we denote

$$L[x] = \{x_j : (x_j, \alpha_j^c, \alpha_j^{(d)}, \xi_j) \in L\}.$$ 

Note that, $L$ is a set of tuples whereas $L[x] \subset \mathbb{R}^n$ is a set of points in $\mathbb{R}^n$.

4.1 Convergence analysis

**Definition 4.1** Let $\{L_k\}$ with $L_k = \{(x_j, \alpha_j, \alpha_j^{(d)}, \xi_j) : j = 1, \ldots, |L_k|\}$ be the sequence of sets of nondominated points produced by DFMOINT. We define a linked sequence as a sequence $\{(x_{jk}, \alpha_{jk}, \alpha_{jk}^{(d)}, \xi_{jk})\}$ such that for any $k = 1, 2, \ldots$, the pair $(x_{jk}, \alpha_{jk}, \alpha_{jk}^{(d)}, \xi_{jk}) \in L_k$ is generated at iteration $k - 1$ of DFMOINT by the pair $(x_{jk-1}, \alpha_{jk-1}, \alpha_{jk-1}^{(d)}, \xi_{jk-1}) \in L_{k-1}$.

Let us, in particular, note the following cases.
Algorithm 1 DFMOINT

1: Let $x_0 \in X \cap \mathbb{Z}$, $\xi_0 > 0$, $\theta \in (0, 1)$, $\epsilon > 0$;
2: let $\{s_k\}$ be a sequence such that $s_k \in S(0, 1)$ for all $k$; let $D = D_0$ be a set of initial primitive discrete directions;
3: let $\alpha_0 = 1$ be the initial stepsize along $s_k$; let $\alpha_i^{(d)} = 1$ be the initial stepsizes along $d \in D$;
4: let $L_0 = \{(x_i, \alpha_i, \alpha_i^{(d)}, \xi_i) : x_i \in X \cap \mathbb{Z}, \alpha_i = \alpha_0, \alpha_i^{(d)} = \alpha_0^{(d)}, i = 1, \ldots, |L_0|\}$
5: For $k = 0, 1, \ldots$
6: Set $\tilde{L}_k = L_k$
7: \textbf{PHASE 1 - Explore continuous variables}
7.1: For $i = 1, 2, \ldots, |L_k^c|$
7.2: Let $(x_i, \alpha_i^c, \alpha_i^{(d)}, \xi_i)$ be the $i$-th tuple of $L_k$
7.3: Compute $\tilde{L}_k$ by the Projected Expansion($x_i, \alpha_i^c, \alpha_i^{(d)}, \xi_i, s_k, \tilde{L}_k$).
7.4: End For
7.5: Set $L_k^c = \tilde{L}_k$ and $L_k^d = L_k^c$
8: \textbf{PHASE 2.A - Explore discrete variables}
8.1: For $j = 1, 2, \ldots, |L_k^d|$
8.2: Let $(x_j, \alpha_j^c, \alpha_j^{(d)}, \xi_j)$ be the $j$-th tuple of $L_k^d$, set $D = D_k$ and $L^+ = L_k^d$
8.3: While $D \neq \emptyset$ and $L^+ = L_k^d[x] = L_k^d[j] = 1$ do
8.4: Choose $d \in D$, set $D = D \setminus \{d\}$
8.5: Compute $\tilde{L}_k$ by the Discrete Search($x_j, \alpha_j^c, \alpha_j^{(d)}, \xi_j, d, L_k^d$).
8.6: End While
8.7: If $L^+ = L_k^d[x] = 1$ for all $d \in D_k$ then
8.8: If $(x_j, \alpha_j^c, \alpha_j^{(d)}, \xi_j) \in L_k^d$ then Set $L_k^d = (L_k^d \setminus \{(x_j, \alpha_j^c, \alpha_j^{(d)}, \xi_j)\}) \cup \{(x_j, \alpha_j^c, \alpha_j^{(d)}, \theta \xi_j)\}$
8.9: End If
8.10: End If
8.11: End For
9: \textbf{PHASE 2.B - Update the set of primitive search directions}
9.1: If $L_k^d = L_k^d$ and $\xi_i$ for all $(x_i, \alpha_i^c, \alpha_i^{(d)}, \xi_i) \in L_k^d$ have been reduced at step 19 then
9.2: If $D_k = D^\circ$ then
9.3: Set $D_{k+1} = D_k$ and $D = D_{k+1}$,
9.4: else
9.5: Generate $D_{k+1}$ such that $D_{k+1} \subseteq D^\circ$ and $D_{k+1} \supseteq D_k$, set $D = D_{k+1}$.
9.6: Set $\alpha_i^{(d)} = 1$ for all $d \in D_{k+1} \setminus D_k$ and $(x_i, \alpha_i^c, \alpha_i^{(d)}, \xi_i) \in L_k^d$
9.7: End If
9.8: else
9.9: Set $D_{k+1} = D_k$ and $D = D_{k+1}$
9.10: End If
9.11: End If
9.12: Set $L_{k+1} = L_k^d$.
9.13: End For
Algorithm 2 Add&Filter \((L, (x, \alpha_x, \alpha_x^{(d)}, \xi_x))\)

Set \(\tilde{L} = \{(x, \alpha_x, \alpha_x^{(d)}, \xi_x)\}\)

For each \((y, \alpha_y, \alpha_y^{(d)}, \xi_y) \in L\)

\[
\text{If } Z(x; \epsilon) \notin Z(y; \epsilon) \text{ then } \tilde{L} = \tilde{L} \cup \{(y, \alpha_y, \alpha_y^{(d)}, \xi_y)\}
\]

End For

Return \(\tilde{L}\).

Algorithm 3 Projected Expansion \((y, \tilde{\alpha}, \alpha^{(d)}, \xi, p, \tilde{L})\)

1: \(\delta, \theta \in (0, 1), \gamma > 0\).
2: Set \(\alpha = \tilde{\alpha}\).
3: If \(\exists x_j \in \tilde{L}: \left(Z([y + \alpha p]_{[l,u]}; \epsilon) > Z(x_j; \epsilon) - \gamma \alpha^2 1\right)\) then
4: \(\text{If } (y, \tilde{\alpha}, \alpha^{(d)}, \xi) \in \tilde{L} \text{ then } \tilde{L} = (\tilde{L} \setminus \{(y, \tilde{\alpha}, \alpha^{(d)}, \xi)\}) \cup \{(y, \theta \tilde{\alpha}, \alpha^{(d)}, \xi)\}\)
5: Return \(\tilde{L}\).
6: End If
7: Let \(\beta = \alpha / \delta\).
8: \(\text{If } \left(Z([y + \beta p]_{[l,u]}; \epsilon) \notin Z([y + \alpha p]_{[l,u]}; \epsilon) - \gamma (\beta^2 - \alpha^2) 1\right)\) then
9: \(\text{Set } \tilde{L} = \text{Add&Filter}(\tilde{L}, ([y + \alpha p]_{[l,u]}, \alpha, \alpha^{(d)}, \xi))\).
10: Endif
11: \(\text{If } \left(Z([y + \beta p]_{[l,u]}; \epsilon) \not\in Z(x_j; \epsilon) - \gamma \beta^2 1\right) \forall x_j \in \tilde{L} \text{ then}\)
12: \(\text{Set } \alpha = \beta \text{ and go to Step 7}\).
13: Endif
14: Return \(\tilde{L}\).

Algorithm 4 Discrete Search \((y, \tilde{\alpha}, \alpha^{(d)}, \xi, p, \tilde{L})\)

1: \(\delta \in (0, 1), \gamma > 0\).
2: Compute the largest stepsize \(\tilde{\alpha}\) s.t. \(y + \tilde{\alpha} p \in X \cap Z\)
3: Set \(\alpha = \min \{\tilde{\alpha}, \alpha_p^{(d)}\}\).
4: If \(\alpha = 0 \text{ or } \left(Z([y + \alpha p]_{[l,u]}; \epsilon) > Z(y; \epsilon) - \xi 1\right)\) then
5: \(\text{Let } \tilde{\alpha}^{(d)} = \alpha^{(d)} \text{ and set } \tilde{\alpha}_p^{(d)} = \max \{1, \lfloor \alpha_p^{(d)}/2 \rfloor\}\)
6: \(\text{If } (y, \tilde{\alpha}, \alpha^{(d)}, \xi) \in \tilde{L} \text{ then } \tilde{L} = (\tilde{L} \setminus \{(y, \tilde{\alpha}, \alpha^{(d)}, \xi)\}) \cup \{(y, \tilde{\alpha}, \tilde{\alpha}^{(d)}, \xi)\}\)
7: Return \(\tilde{L}\).
8: End If
9: Let \(\beta = \min \{\tilde{\alpha}, 2 \alpha_p^{(d)}\}\).
10: \(\text{If } \left(Z([y + \beta p]_{[l,u]}; \epsilon) \not\in Z([y + \alpha p]_{[l,u]}; \epsilon) - \xi 1\right)\) then
11: \(\text{Let } \tilde{\alpha}^{(d)} = \alpha^{(d)} \text{ and set } \tilde{\alpha}_p^{(d)} = \alpha\)
12: \(\text{Set } \tilde{L} = \text{Add&Filter}(\tilde{L}, ([y + \alpha p], \tilde{\alpha}, \tilde{\alpha}^{(d)}, \xi))\).
13: Endif
14: If \(\beta < \tilde{\alpha} \text{ and } \left(Z([y + \beta p]_{[l,u]}; \epsilon) \not\in Z(x_j; \epsilon) - \xi 1\right) \forall x_j \in \tilde{L} \text{ then}\)
15: \(\text{Set } \alpha = \beta \text{ and go to Step 9}\).
16: Endif
17: If \(\beta = \tilde{\alpha} \text{ then}\)
18: \(\text{Let } \tilde{\alpha}^{(d)} = \alpha^{(d)} \text{ and set } \tilde{\alpha}_p^{(d)} = \beta\)
19: \(\text{Set } \tilde{L} = \text{Add&Filter}(\tilde{L}, ([y + \beta p], \tilde{\alpha}, \tilde{\alpha}^{(d)}, \xi))\)
20: Endif
21: Return \(\tilde{L}\).
1. (Success w.r.t. continuous variables) we have

\[ x_{jk} = \left[ x_{jk-1} + \frac{\alpha_{jk-1} d_{k-1}}{\delta s_{k-1}} \right]_{[l,u]}, s_{k-1} \geq 0; \]

\[ \alpha_{jk} = \frac{\alpha_{jk-1}}{\delta s_{k-1}} \]

\[ \alpha^{(d)}_{jk} = \alpha^{(d)}_{jk-1} \]

\[ \xi_{jk} = \xi_{jk-1} \]

and it holds that

\[ Z \left( \left[ x_{jk-1} + \frac{\alpha_{jk-1} d_{k-1}}{\delta s_{k-1}} \right]_{[l,u]} ; \epsilon \right) \prec Z \left( \left[ x_{jk-1} + \frac{\alpha_{jk-1} d_{k-1}}{\delta s_{k-1}} \right]_{[l,u]} ; \epsilon \right) \]

\[ -\gamma \left( \left( \frac{\alpha_{jk-1}}{\delta s_{k-1}} \right)^2 - \left( \frac{\alpha_{jk-1}}{\delta s_{k-1}} \right)^2 \right) \mathbf{1}. \] (18)

2. (Failure w.r.t. continuous variables) we have

\[ Z \left( [x_{jk-1} + \alpha_{jk-1} d_{k-1}]_{[l,u]} ; \epsilon \right) > Z(x_{\ell_j} ; \epsilon) - \gamma (\alpha_{jk-1})^2 \mathbf{1}, \] (19)

for at least a point \( x_{\ell_j} \in \tilde{L}_{k-1} \), that is, \( Z([x_{jk-1} + \alpha_{jk-1} d_{k-1}]_u ; \epsilon) \) is dominated by \( Z(x_{\ell_j} ; \epsilon) - \gamma (\alpha_{jk-1})^2 \mathbf{1} \). This implies that \([x_{jk-1} + \alpha_{jk-1} d_{k-1}]_u \) is a “bad” point so that the step \( \alpha_{jk-1} \) associated to \( x_{jk-1} \) is shrunk by the constant factor \( \theta \), i.e.

\[ x_{jk} = x_{jk-1} \] \hspace{1cm} (20a)

\[ \alpha_{jk} = \theta \alpha_{jk-1} \] \hspace{1cm} (20b)

\[ \alpha^{(d)}_{jk} = \alpha^{(d)}_{jk-1} \] \hspace{1cm} (20c)

\[ \xi_{jk} = \xi_{jk-1} \] \hspace{1cm} (20d)

3. (Success w.r.t. discrete variables) we have

\[ x_{jk} = \left( x_{jk-1} + \alpha^{(d)}_{jk-1} p_{k-1} \right); \]

\[ \alpha_{jk} = \alpha_{jk-1} \]

\[ \alpha^{(d)}_{jk} = \alpha^{(d)}_{jk-1} \]

\[ \xi_{jk} = \xi_{jk-1} \]

and it holds that

\[ Z \left( x_{jk-1} + \beta p_{k-1} ; \epsilon \right) \prec Z \left( x_{jk-1} + \alpha^{(d)}_{jk-1} p_{k-1} ; \epsilon \right) - \xi \mathbf{1} \] \hspace{1cm} (21)

\[ \beta \leq 2 \alpha^{(d)}_{jk-1} \]

4. (Failure w.r.t. discrete variables) we have

\[ Z([x_{jk-1} + p_{k-1}]_u ; \epsilon) > Z(x_{\ell_j} ; \epsilon) - \xi \mathbf{1}, \] (22)

for at least a point \( x_{\ell_j} \in \tilde{L}_{k-1} \) and all \( p_{k-1} \in D_{k-1} \). This implies

\[ x_{jk} = x_{jk-1} \] \hspace{1cm} (23a)

\[ \alpha_{jk} = \alpha_{jk-1} \] \hspace{1cm} (23b)

\[ \alpha^{(d)}_{jk} = \alpha^{(d)}_{jk-1} = 1 \] \hspace{1cm} (23c)

\[ \xi_{jk} = \theta \xi_{jk-1} \] \hspace{1cm} (23d)
Lemma 4.2 ([24, Lemma 3.15]) Let \( \alpha, \beta \in \mathbb{R} \), with \( \alpha, \beta \neq 0 \), be such that \( \text{sign}(\alpha) = \text{sign}(\beta) \) and let \( x \in X \) and \( p \in \mathbb{R}^n \). Then, we have
\[
[x + (\alpha + \beta)p]_{[l,u]} = [x + \alpha p]_{[l,u]} + \beta p]_{[l,u]}
\]

Proposition 4.3 The Projected Expansion is well-defined, that is, either the test at step 3 is satisfied or steps 7–12 are executed. In the latter case:

i) the test at Step 11 is satisfied a finite number of times, i.e. the procedure cannot infinitely cycle;

ii) the procedure updates \( \tilde{L} \), i.e. the test at Step 8 is satisfied at least once, i.e. Add\&Filter is called at least once.

Proof. If the test at step 3 is satisfied then, the procedure terminates. On the other hand, namely if the test at step 3 is not satisfied, the procedure executes steps 7–12 thus becoming equivalent to the Projected expansion in [24]. Hence, in the latter case, the proof follows from the proof of Proposition 3.16 of [24]. \( \square \)

In the following we prove an analogous result for the Discrete Search procedure.

Proposition 4.4 The Discrete Search is well-defined, that is, either the test at step 4 is satisfied or steps 9–15 are executed. In the latter case:

i) the test at Step 14 is satisfied a finite number of times, i.e. the procedure cannot infinitely cycle;

ii) the procedure updates \( \tilde{L} \), i.e. either Step 12 or 19 are executed, that is Add\&Filter is called at least once.

Proof. If the test at step 4 is satisfied, the procedure terminates. On the other hand, steps 9-15 get executed.

Point i). We proceed by contradiction and assume that the test at Step 14 is always satisfied, i.e. a monotonically increasing sequence of positive numbers \( \{ \beta^j \} \) exists. But this contradicts the fact that, by compactness of set \( X \cap Z \), \( \bar{\alpha} \) is finite.

Point ii). If \( \beta = \bar{\alpha} \), then the Discrete Search updates \( \tilde{L} \) at step 19 which concludes the proof. Otherwise, let us suppose that \( \beta < \bar{\alpha} \) at every iteration of the Discrete Search. In this case, let us consider a generic iteration. Then, we have
\[
Z([y + \alpha p]_{[l,u]}; \epsilon) < Z([y + \alpha p]_{[l,u]}; \epsilon) - \xi
\]
for all \( x_j \in \tilde{L} \), either because \( \alpha = \alpha_p^{(d)} \) or because the test at step 14 is satisfied and \( \alpha = \beta \). The above relation imply that, for all \( x_j \in \tilde{L} \), and index \( \ell_j \in \{ 1, \ldots, q \} \) exists such that
\[
Z_{\ell_j}([y + \alpha p]_{[l,u]}; \epsilon) \leq Z_{\ell_j}(x_j; \epsilon) - \xi.
\]
Furthermore, let us assume that the test at Step 10 is not satisfied, which means that
\[
Z([y + \beta p]_{[l,u]}; \epsilon) < Z([y + \alpha p]_{[l,u]}; \epsilon) - \xi \]
and, in particular, in view of (24), for all \( x_j \in \tilde{L} \), and index \( \ell_j \in \{ 1, \ldots, q \} \) exists such that
\[
Z_{\ell_j}([y + \beta p]_{[l,u]}; \epsilon) < Z_{\ell_j}([y + \alpha p]_{[l,u]}; \epsilon) - \gamma \xi \leq Z_{\ell_j}(x_j; \epsilon) - 1.
\]
The above relation, recalling that \( \beta < \bar{\alpha} \), implies that the test at Step 14 is satisfied so that the procedure will perform a further iteration.

Now, let us assume by contradiction that the test at Step 10 is never satisfied. Then, the above reasoning would imply that the procedure infinitely cycles, which contradicts Point i). \( \square \)
Proposition 4.5 Let Assumption 2 hold. Let \( \{L_k\} \) with \( L_k = \{(x_j, \alpha_j), j = 1, \ldots, |L_k|\} \) be the sequence of sets of nondominated pairs produced by Algorithm DFMOINT. Let \( \{(x_{j_k}, \alpha_{j_k})\} \) be a linked sequence and \( \bar{x} \) be any limit point of \( \{x_{j_k}\} \), i.e.,

\[
\lim_{k \to \infty, k \in K} x_{j_k} = \bar{x},
\]

for a subset \( K \) of indices. Then, an \( \epsilon^* > 0 \) exists such that for all \( \epsilon \in (0, \epsilon^*) \), if the subsequence \( \{d_k\}_{k \in K} \) is dense in the unit sphere, \( \bar{x} \) is a Pareto-Clarke KKT stationary point for Problem (1) w.r.t. the continuous variables.

Proof. If \( \lim_{k \to \infty, k \in K} x_{j_k} = \bar{x} \), an index \( \bar{k} \) exists such that, for all \( k \geq \bar{k}, k \in K \), \( (x_{j_k})_z = (\bar{x})_z \).

Then, the proof follows from the proof of [24, Proposition 3.19]. \( \square \)

Proposition 4.6 Let \( \{L_k\} \) with \( L_k = \{(x_j, \alpha_j, \alpha_j^{(d)}; \xi_j), j = 1, \ldots, |L_k|\} \) be the sequence of sets of nondominated points produced by DFMOINT. Every linked sequence \( \{(x_{j_k}, \alpha_{j_k}, \alpha_{j_k}^{(d)}, \xi_{j_k})\} \) is such that

\[
\lim_{k \to \infty} \xi_{j_k} = 0.
\]

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

\[
\xi_{j_{k+1}} = \xi_{j_k} = M
\] \quad (25)

for all \( k \geq \bar{k} \). Now, we preliminary show that, for every primitive direction \( p \), a \( \bar{k} \) exists such that, for all \( k \geq \bar{k}, \alpha_{p,j_k}^{(d)} = 1 \). We again proceed by contradiction and assume that and index set \( K \subset \{0, 1, 2, \ldots\} \) exists such that

\[
\alpha_{p,j_k}^{(d)} > 1, \quad \text{for all} \quad k \in K.
\]

If this is the case, for \( k \in K \), the test at step 4 of the Discrete Search procedure is not satisfied. This means that \( \alpha > 0 \) and, for all \( x_j \in L_k \), \( Z(x_{j_k} + \alpha p; \epsilon) \neq Z(x_{j_k}; \epsilon) - \xi_{j_k} \). Hence, an index \( \ell_k \in \{1, \ldots, q\} \), exists such that

\[
Z_{\ell_k}(x_{j_k} + \alpha p; \epsilon) \leq Z_{\ell_k}(x_{j_k}; \epsilon) - \xi_{j_k} \leq Z_{\ell_k}(x_{j_k}; \epsilon) - M.
\]

Since, the number of objective functions is finite, it is possible to extract a further subset \( K \subseteq K \) such that, \( \ell_k = \hat{\ell} \), for all \( k \in K \) and

\[
Z_{\ell}(x_{j_k} + \alpha p; \epsilon) \leq Z_{\ell}(x_{j_k}; \epsilon) - \xi_{j_k} \leq Z_{\ell}(x_{j_k}; \epsilon) - M.
\]

The above relation is in contrast with continuity assumption of \( Z_{\ell} \) and compactness of \( X \cap Z \).

Now, by compactness of \( X \cap Z \), the feasible primitive directions are finite. Hence, an index \( \hat{k} \) exists such that

\[
\alpha_{p,j_k}^{(d)} = 1, \quad \text{for all} \quad p \in D, \quad k \geq \hat{k}.
\]

Hence, for \( k \geq \hat{k} \), the Discrete Search does not change the points within \( L_{j_k}^{d} \), i.e. \( L_{j_k}^{+}[x] = L_{j_k}^{d}[x] \). In this case, by the instruction of the algorithm DFMOINT, \( \xi_{j_{k+1}} = \theta \xi_{j_k} = \theta M < \xi_{j_k} \) which is a contradiction with (25). \( \square \)

For every linked sequence \( \{(x_{j_k}, \alpha_{j_k}, \alpha_{j_k}^{(d)}, \xi_{j_k})\} \), let us define the set

\[
H = \{k : \xi_{j_{k+1}} < \xi_{j_k}\}.
\]
Proposition 4.7 Let Assumption 2 hold. Let \( \{L_k\} \) with \( L_k = \{(x_j, \alpha_j), j = 1, \ldots, |L_k|\} \) be the sequence of sets of nondominated pairs produced by Algorithm DFMOINT. Let \( \{(x_{jk}, \alpha_{jk}, \alpha_{jk}^{(d)}, \xi_{jk})\} \) be a linked sequence and \( \bar{x} \) be any limit point of \( \{x_{jk}\}_H \), i.e., a index \( K \subseteq H \) exists such that
\[
\lim_{k \to \infty, k \in K} x_{jk} = \bar{x},
\]
Then, an \( \epsilon^* > 0 \) exists such that for all \( \epsilon \in (0, \epsilon^*], \bar{x} \) is a Pareto-Clarke KKT stationary point for Problem (1).

Proof. By the fact that \( k \in K \), we know that an index \( \bar{k} \in K \) exists such that
\[
(x_{jk})_z = (x_{jk+1})_z = \bar{x}_z, \quad \forall \ k \in K, k \geq \bar{k},
\]
\[
(\alpha_{jk}^{(d)})_p = 1, \quad \forall \ p \in D_k
\]
Furthermore, since \( \alpha_{jk}^c \to 0 \), we also have that
\[
\lim_{k \to \infty, k \in K} \alpha_{jk+1}^c = 0.
\]
So that
\[
\lim_{k \to \infty, k \in K} (x_{jk+1})^c = (\bar{x})^c,
\]
hence we get
\[
\lim_{k \to \infty, k \in K} x_{jk+1} = \bar{x}.
\]
Let us consider any point \( \hat{x} \in B^z(\bar{x}) \). By the definition of discrete neighborhood \( B^z(\bar{x}) \), a direction \( \hat{d} \in D^z(\bar{x}) \) exists such that
\[
\hat{x} = \bar{x} + \hat{d}.
\]
Then, for all \( k \in K \) sufficiently large, (26) and (27) imply
\[
(x_{jk} + \hat{d})_z = (x_{jk+1} + \hat{d})_z = (\bar{x} + \hat{d})_z = (\hat{x})_z.
\]
Hence, for all \( k \in K \) sufficiently large, by the definition of discrete neighborhood we have \( \hat{d} \in D^z(x_{jk+1}) \) and
\[
x_{jk} + \hat{d} \in X \cap Z.
\]
Then, since \( k \in K \subseteq H \), by the definition of \( H \) and by the instructions of the Discrete search procedure, we have
\[
Z(x_{jk} + \hat{d}; \epsilon) > Z(x_{jk}) - \xi_{jk}.
\]
By taking the limit for \( k \to \infty, k \in K \), we get
\[
Z(\hat{x}; \epsilon) \geq Z(\bar{x}),
\]
hence \( \hat{x} \) cannot dominate \( \bar{x} \) and the result follows by Proposition 4.5.

5 Numerical Results

In this section we report the results obtained in the numerical experimentation and comparison of the proposed algorithm with state-of-the-art methods in the literature on a set of multiobjective test problems.
Test problems description
We used the 10 multiobjective problems proposed for the CEC 2009 special session and competition [28]. These problems have \( n = 10 \) variables and two or three objective functions. Since the number of variables can be modified for all of the problems, we considered \( n \in \{10, 15, 20, 25, 30 \} \) thus obtaining a total of 50 problems.

In order to come up with mixed integer problems, given a continuous problem (among the 50 above defined)

\[
\begin{align*}
\min & \ f_1(\tilde{x}), \ldots, f_q(\tilde{x}) \\
\text{s.t.} & \ \ell_i \leq \tilde{x}_i \leq u_i, \ \forall \ i = 1, \ldots, n
\end{align*}
\]

we consider \( n_c = \lfloor n/2 \rfloor \) continuous variables and, consequently, \( n_z = n - n_c \) integer variables. Hence, we define the following discretized problem

\[
\begin{align*}
\min & \ f_1(x), \ldots, f_q(x) \\
\text{s.t.} & \ \ell_i \leq x_i \leq u_i, \ \forall \ i = 1, \ldots, n_c \\
& 0 \leq x_i \leq 100, \ \forall \ i = n_c + 1, \ldots, n
\end{align*}
\]  \hspace{1cm} (28)

where

\[
\begin{align*}
\tilde{x}_i &= x_i, \quad i = 1, \ldots, n_c, \\
\tilde{x}_i &= \ell_i + \frac{u_i - \ell_i}{100} x_i, \quad i = n_c + 1, \ldots, n
\end{align*}
\]

and \( f_j(x) = f_j(\tilde{x}), \ j = 1, \ldots, q \).

As concerns constrained problems, given a bound constrained problem like problem [28], we generate six constrained problems by adding the families of nonlinear constraints proposed in [20] and reported in table 1. Hence, we come up with 300 nonlinearly constrained multiobjective problems

\[
\begin{align*}
g_j(x) &= (3 - 2x_{j+1})x_{j+1} - x_j - 2x_{j+2} + 1 \leq 0, \quad \text{for all } j \in \{1, 2, \ldots, n - 2 \}, \\
g_j(x) &= (3 - 2x_{j+1})x_{j+1} - x_j - 2x_{j+2} + 2.5 \leq 0, \quad \text{for all } j \in \{1, 2, \ldots, n - 2 \}, \\
g_j(x) &= x_j^2 + x_{j+1}^2 + x_jx_{j+1} - 2x_j - 2x_{j+1} + 1 \leq 0, \quad \text{for all } j \in \{1, 2, \ldots, n - 1 \}, \\
g_j(x) &= x_j^2 + x_{j+1}^2 + x_jx_{j+1} - 1 \leq 0, \quad \text{for all } j \in \{1, 2, \ldots, n - 1 \}, \\
g_j(x) &= (3 - 0.5x_{j+1})x_{j+1} - x_j - 2x_{j+2} + 1 \leq 0, \quad \text{for all } j \in \{1, 2, \ldots, n - 2 \}, \\
g_j(x) &= \sum_{j=1}^{n-2}((3 - 0.5x_{j+1})x_{j+1} - x_j - 2x_{j+2} + 1) \leq 0.
\end{align*}
\]

Table 1: Families of constraints from [20].

with number of variables \( n \in \{10, 15, 20, 25, 30 \} \), number of objective functions \( q \in \{2, 3 \} \) and number of constraints \( m \in \{1, 8, 9, 13, 14, 18, 19, 23, 24, 28, 29 \} \).

Methods
The proposed method DFMOINT implemented in python 3.9 (along with the test problems) is freely and publicly available on the Derivative-Free Library DFL at the URL [http://www.iasi.cnr.it/~liuzzi/DFL/](http://www.iasi.cnr.it/~liuzzi/DFL/)

In the experimentation, we compared DFMOINT with two representative methods of, respectively, deterministic and probabilistic algorithms for mixed-integer multiobjective black-box optimization problems. Namely, we considered:

- the BiMADS algorithm for bi-objective problems implemented in the NOMAD (version 3.9.1) software package [21][1][2];
- the NSGA-II algorithm [14] implemented in the pygmo python library.
All of the methods were run by using their respective default parameters and allowing a maximum number of 20,000 function evaluations. In particular, NSGA-II was run using a population size of 100 individuals and allowing 200 generations so that NSGA-II uses exactly 20,000 function evaluations.

Since NSGA-II is a probabilistic algorithm, its outcomes vary if it is run using different seeds for the random sequence. Hence, we run NSGA-II 10 times with ten different random seed. Then, on the basis of the results obtained by these ten runs, we extracted the best, median and worst (ideal) algorithms for every single problem. The best run simply consists of the run that has the higher percentage of nondominated solutions when compared to the remaining ones (considering as a reference Pareto front the one obtained from the 10 runs performed). In a similar way, the worst run is selected as the one with the lowest percentage of nondominated points.

**Metrics and performance profiles**

The results of the various comparisons are reported in terms of the purity $\mathcal{G}$, $\Gamma$ and $\Delta$ spread metrics (as defined in [11]) by using performance profiles [15].

**Results on bound constrained problems**

In figure 1, we report the comparison of the proposed algorithm DFMOINT with the three versions of NSGA-II, namely the best, worst and median. The results are reported in terms of performance profiles. As we can see, DFMOINT compares favorably with NSGA-II (median) in terms of purity. In particular, DFMOINT is more efficient than NSGA-II (median) but slightly less robust. As regards, the spread $\Gamma$ metric, all of the algorithms have very similar performances whereas, for the spread $\Delta$ metric DFMOINT behaves poorly with respect to NSGA-II. If we consider that we are comparing our algorithm against NSGA-II which is designed to perform a global exploration of the search space, the comparison is not that bad. In fact, the following observations can be carried out:

(i) in terms of purity, DFMOINT performances are closer to the best version of NSGA-II rather than to the worst version; furthermore, DFMOINT is competitive with the median version of NSGA-II;

(ii) the behaviour of DFMOINT with respect to the spread metrics was largely expected given the local optimization nature of our proposed algorithm and again considering the definition of NSGA-II.

Concerning the comparison with BiMADS, we have to first remark that BiMADS can only solve problems with only two objective functions, whereas DFMOINT is able to manage an arbitrary number of them. Hence, the comparison has been carried out on the subset of problems with $q = 2$ objective functions, i.e. 35 problems out of the 50 bound constrained problems. The results are reported in figure 2. DFMOINT compares favorably with BiMADS in terms of spread $\Delta$ and is the clear winner w.r.t. the spread $\Gamma$. For what concerns purity, it can be noted that BiMADS is more efficient than DFMOINT (on the subset of two-objective problems) but the two methods have the same robustness. It must also be said that BiMADS has been run using default values for the parameters. In particular, the scalarized problems are solved with MADS using Nelder-Meade heuristic in the initial phase of the optimization process.

**Results on constrained problems**

For the constrained problems set, it must be said that the adding of constraints reported in table 1 can conflict with the bound constraints on the variables. Indeed, we noticed that for some problems no solver is able to find feasible non-dominated points. Hence, in the comparison we keep only those problems where at least one solver is able to find feasible non-dominated points.
As concerns the behavior of DFMOINT on the nonlinearly contrained problems, figure 3 reports the comparison of DFMOINT with respect to the three versions of NSGA-II. We can make the following observations:

(i) in terms of purity, DFMOINT is better than NSGA-II median and worst and it is slightly dominated by the best version of NSGA-II;

(ii) in terms of spread $\Gamma$, DFMOINT is the clear winner with respect to all of the versions of NSGA-II;

(iii) in terms of spread $\Delta$, NSGA-II (each version) is more efficient than DFMOINT but less robust.

More or less the same observations can be made with regards to the comparison between DFMOINT and BiMADS on the subset of the 210 bi-objective constrained problems where at least one solver finds feasible non-dominated points. The profiles for such a comparison are reported in figure 4. As we can see, DFMOINT is better than BiMADS with respect to all of the three metrics considered, namely purity, spread $\Gamma$ and $\Delta$. 
Figure 2: Comparison of DFMOINT with BiMADS (implemented in NOMAD 3.9.1) on the 35 bound constrained problems with $q = 2$ objective functions in terms of purity, spread $\Gamma$ and $\Delta$ performance profiles.

6 Conclusions

In this paper we proposed a new derivative-free algorithm for the solution of mixed-integer black-box constrained multiobjective optimization problems. In particular, considering the presence of integer or discrete variables and the possible nonsmoothness of the functions defining the problem, we give local/global Pareto optimality definitions and stationarity conditions. Then, through the use of an exact penalty approach we show the equivalence between the original nonlinearly constrained multiobjective problem and a penalized problem where only bound constraints are present. For the proposed algorithm, we carry out a thorough convergence analysis and we manage to prove convergence toward stationary points of the problem. The proposed algorithm (DFMOINT) has been tested on a large number of bound constrained problems and nonlinearly constrained problems and its performances have been compared with other state-of-the-art solvers for mixed-integer derivative-free multiobjective optimization. In particular, we considered in the experimentation two well-known solvers, namely BiMADS (implemented in the software package NOMAD v3.9.1) and NSGA-II (implemented in the pygmo python library).
The results and comparison show the efficiency of DFMOINT with respect to both probabilistic and deterministic approaches.

References

[1] C. Audet, S. Le Digabel, C. Tribes, and V. Rochon Montplaisir. The nomad project. software available at https://www.gerad.ca/nomad.

[2] C. Audet, G. Savard, and W. Zghal. Multiobjective optimization through a series of single-objective formulations. *SIAM Journal on Optimization*, 19(1):188–210, 2008.

[3] S. Bandyopadhyay, S.K. Pal, and B. Aruna. Multiobjective GAs, quantitative indices, and pattern classification. *Systems, Man, and Cybernetics, Part B: IEEE Transactions on Cybernetics*, 34(5):2088–2099, 2004.
Figure 4: Comparison of DFMOINT with BiMADS (implemented in NOMAD 3.9.1) on the subset of 210 bi-objective constrained problems where at least one solver finds feasible nondominated points in terms of purity, spread $\Gamma$ and $\Delta$ performance profiles.

[4] Jean Bigeon, Sébastien Le Digabel, and Ludovic Salomon. Dmulti-mads: mesh adaptive direct multisearch for bound-constrained blackbox multiobjective optimization. *Computational Optimization and Applications*, 79(2):301–338, 2021.

[5] Carmo P Brás and Ana Luísa Custódio. On the use of polynomial models in multiobjective directional direct search. *Computational Optimization and Applications*, 77(3):897–918, 2020.

[6] Emilio Fortunato Campana, Matteo Diez, Giampaolo Liuzzi, Stefano Lucidi, Riccardo Pellegrini, Veronica Piccialli, Francesco Rinaldi, and Andrea Serani. A multi-objective direct algorithm for ship hull optimization. *Computational optimization and applications*, 71(1):53–72, 2018.

[7] F.H. Clarke. *Optimization and Nonsmooth Analysis*. John Wiley & Sons, New York, 1983.

[8] Frank H Clarke. *Optimization and nonsmooth analysis*. SIAM, 1990.
[9] Guido Cocchi, Giampaolo Liuzzi, Anthony Papini, and Marco Sciandrone. An implicit filtering algorithm for derivative-free multiobjective optimization with box constraints. *Computational Optimization and Applications*, 69(2):267–296, 2018.

[10] A. L. Custódio, M. Emmerich, and J.F.A. Madeira. Recent developments in derivative-free multiobjective optimization. *Computational Technology Reviews*, 5:1–30, 2012.

[11] A.L. Custódio, J.F.A. Madeira, A.I.F. Vaz, and L.N. Vicente. Direct multisearch for multiobjective optimization. *SIAM Journal on Optimization*, 21(3):1109–1140, 2011.

[12] Ana Luísa Custódio, Youssef Diouane, Rohollah Garanjanji, and Elisa Riccietti. Worst-case complexity bounds of directional direct-search methods for multiobjective optimization. *Journal of Optimization Theory and Applications*, 188(1):73–93, 2021.

[13] Ana Luísa Custódio and JF Aguilar Madeira. Multiglods: global and local multiobjective optimization using direct search. *Journal of Global Optimization*, 72(2):323–345, 2018.

[14] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

[15] E.D. Dolan and J.J. Moré. Benchmarking optimization software with performance profiles. *Mathematical programming*, 91(2):201–213, 2002.

[16] G. Fasano, G. Liuzzi, S. Lucidi, and F. Rinaldi. A linesearch-based derivative-free approach for nonsmooth constrained optimization. *SIAM Journal on Optimization*, 24(3):959–992, 2014.

[17] E.H. Fukuda, L.M. Graña Drummond, and F.M.P. Raupp. An external penalty-type method for multicriteria. *TOP*, 24(2):493–513, 2016.

[18] X.X. Huang and X.Q. Yang. Nonlinear lagrangian for multiobjective optimization and applications to duality and exact penalization. *SIAM Journal on Optimization*, 13(3):675–692, 2002.

[19] J. Jahn. *Introduction to the theory of nonlinear optimization*. Springer-Verlag, Berlin, 1996.

[20] N. Karmitsa. Test problems for large-scale nonsmooth minimization. Technical Report No. B. 4/2007, Department of Mathematical Information Technology, University of Jyväskylä, 2007.

[21] Sébastien Le Digabel. Algorithm 909: Nomad: Nonlinear optimization with the mads algorithm. *ACM Transactions on Mathematical Software (TOMS)*, 37(4):1–15, 2011.

[22] Chih-Jen Lin, Stefano Lucidi, Laura Palagi, Arnaldo Risi, and Marco Sciandrone. Decomposition algorithm model for singly linearly-constrained problems subject to lower and upper bounds. *Journal of Optimization Theory and Applications*, 141(1):107–126, 2009.

[23] Giampaolo Liuzzi, Stefano Lucidi, and Veronica Piccialli. Exploiting derivative-free local searches in direct-type algorithms for global optimization. *Computational Optimization and Applications*, 65(2):449–475, 2016.

[24] Giampaolo Liuzzi, Stefano Lucidi, and Francesco Rinaldi. A derivative-free approach to constrained multiobjective nonsmooth optimization. *SIAM Journal on optimization*, 26(4):2744–2774, 2016.

[25] Julia Niebling and Gabriele Eichfelder. A branch–and–bound–based algorithm for nonconvex multiobjective optimization. *SIAM Journal on Optimization*, 29(1):794–821, 2019.

[26] R. Tyrrel Rockafellar. *Convex Analysis*. Princeton University Press, 1970.
[27] A. Suppapitnarm, K.A. Seffen, G.T. Parks, and P.J. Clarkson. A simulated annealing algorithm for multiobjective optimization. *Engineering Optimization*, 33(1):59–85, 2000.

[28] Qingfu Zhang, Aimin Zhou, Shizheng Zhao, Ponnuthurai Nagaratnam Suganthan, Wudong Liu, Santosh Tiwari, et al. Multiobjective optimization test instances for the cec 2009 special session and competition (https://esa.github.io/pagmo2/docs/cpp/problems/cec2009.html). *University of Essex, Colchester, UK and Nanyang technological University, Singapore, special session on performance assessment of multi-objective optimization algorithms, technical report*, 264:1–30, 2008.