Les Cahiers du GERAD ISSN: 0711–2440

Adaptive direct search algorithms with relaxable and quantifiable constraints

C. Audet, T. Denorme, Y. Diouane, S. Le Digabel, C. Tribes

G–2026–39

July 2026

Citation suggérée : C. Audet, T. Denorme, Y. Diouane, S. Le Digabel, C. Tribes (Juillet 2026). Adaptive direct search algorithms with relaxable and quantifiable constraints, Rapport technique, Les Cahiers du GERAD G– 2026–39, GERAD, HEC Montréal, Canada.

Avant de citer ce rapport technique, veuillez visiter notre site Web (https://www.gerad.ca/fr/papers/G-2026-39) afin de mettre à jour vos données de référence, s'il a été publié dans une revue scientifique.

Suggested citation: C. Audet, T. Denorme, Y. Diouane, S. Le Digabel, C. Tribes (July 2026). Adaptive direct search algorithms with relaxable and quantifiable constraints, Technical report, Les Cahiers du GERAD G–2026–39, GERAD, HEC Montréal, Canada.

Before citing this technical report, please visit our website (https://www.gerad.ca/en/papers/G-2026-39) to update your reference data, if it has been published in a scientific journal.

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2026
– Bibliothèque et Archives Canada, 2026

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2026
– Library and Archives Canada, 2026

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Adaptive direct search algorithms with relaxable and quantifiable constraints

Charles Audet

Théo Denorme

Youssef Diouane

Sébastien Le Digabel

Christophe Tribes

GERAD, Montréal (Qc), Canada, H3T 1J4

Département de mathématiques et génie industriel, Polytechnique Montréal, Montréal (Qc), Canada, H3C 3A7

charles.audet@polymtl.ca
theo.denorme@polymtl.ca
youssef.diouane@polymtl.ca
sebastien.le-digabel@polymtl.ca
christophe.tribes@polymtl.ca

July 2026
Les Cahiers du GERAD
G–2026–39

Abstract : This work introduces ADS-PB, an extension of the Adaptive Direct Search (ADS) framework for solving constrained blackbox optimization problems. With ADS, iterates progress without relying on mesh structures or sufficient decrease conditions on the objective function value. Unlike the extreme barrier approach used in ADS, where only unrelaxable constraints are considered, the proposed method also handles quantifiable and relaxable constraints using a Progressive Barrier (PB) mechanism that exploits both constraint and objective function values. A convergence analysis of the proposed framework under mild assumptions is presented. The performance of the proposed method is assessed using sets of analytical and simulation-based constrained test problems and is compared with state-of-the-art blackbox optimization solvers, including the PB approach within the Mesh Adaptive Direct Search (MADS) framework.

Keywords : Derivative-free optimization; constrained optimization; adaptive direct search; eelaxable constraints

AMS subject classifications : 90C30, 90C56, 49J52.

Résumé : Ce travail introduit ADS-PB, une extension du cadre de recherche directe adaptative (ADS) pour la résolution de problèmes d'optimisation de boîte noire sous contraintes. Avec ADS, les itérés progressent sans s'appuyer sur des structures de maillage ni sur des conditions de décroissance suffisante de la valeur de la fonction objectif. Contrairement à l'approche de barrière extrême utilisée dans ADS, où seules les contraintes non relaxables sont considérées, la méthode proposée traite également les contraintes quantifiables et relaxables au moyen d'un mécanisme de barrière progressive (PB) qui exploite à la fois les valeurs des contraintes et de la fonction objectif. Une analyse de convergence du cadre proposé sous des hypothèses faibles est présentée. Les performances de la méthode sont évaluées sur des ensembles de problèmes tests contraints analytiques et basés sur des simulations, et comparées à des solveurs d'optimisation de boîte noire de l'état de l'art, y compris l'approche PB au sein du cadre de recherche directe adaptative sur maillage (MADS).

Mots clés : optimisation sans dérivées ; optimisation sous contraintes ; recherche directe adaptative ; contraintes relaxables

Acknowledgements: This research is funded by the NSERC discovery grants RGPIN–2026–05340 (Audet), RGPIN–2024–05093 (Diouane) and RGPIN–2024–05086 (Le Digabel).

1 Introduction

Constrained blackbox optimization arises in many applications where both the objective function and the constraints can only be accessed through evaluations, often at a significant computational cost and without derivative information. In such settings, the optimization framework must not only remain robust in the absence of smoothness assumptions but also be flexible enough to exploit informative trial points proposed by search heuristics, surrogate models, or problem-specific strategies. The present work considers constrained blackbox optimization problems [11] of the form

$$\min_{x\in\Omega} f(x) \tag{1}$$

where the feasible set is defined as $\Omega = \{x \in X : c_j(x) \leq 0 \text{ for all } j \in J\}$, and $X \subseteq \mathbb{R}^n$ denotes the blackbox domain. Using the taxonomy defined in [35], the set X incorporates the *a priori known* [35] constraints of the problem, such as bound constraints or other explicitly defined restrictions. The index set J is finite and identifies the inequality constraints. Both the objective function $f\colon X \to \overline{\mathbb{R}}$ and the constraint functions $\{c_j\colon X \to \overline{\mathbb{R}}\}_{j\in J}$ are assumed to be available only through blackbox evaluations. The use of the extended real line $\overline{\mathbb{R}}$ instead of $\mathbb{R}$ reflects the fact that, in blackbox settings, some evaluations may fail or return invalid outputs. For instance, a simulation may crash, a solver may not converge, or a function call may return an error at some trial points. In such cases, assigning the value $+\infty$ to the objective and/or constraint functions provides a convenient way to model these failed evaluations while keeping the optimization framework robust to this type of numerical issue. The constraints are assumed to be *quantifiable* and *relaxable* [35]. They are said to be *quantifiable* because each constraint function c_j, $j \in J$, provides a real-valued measure of violation, allowing one to assess not only feasibility but also the magnitude of infeasibility. Such constraints are said to be *relaxable* because the objective may still be evaluated at infeasible points during the optimization process and used to guide the search toward feasibility, rather than being strictly excluded from the search.

1.1 Motivation

The present work extends the Adaptive Direct Search (ADS) framework of [10] to the class of constrained blackbox optimization problems of the form (1). While ADS relies on an extreme barrier [8] to enforce feasibility, such an approach can be overly restrictive when constraint violations can be quantified and meaningfully reduced during the optimization process. This naturally motivates the use of a Progressive Barrier (PB) mechanism [9], which allows infeasible points to contribute useful information while gradually driving the optimization process toward feasibility.

More broadly, many established Derivative-Free Optimization (DFO) methods have been proposed for constrained problems, including evolution strategies [25, 30], a merit function approach for direct search [28], a mixed interior point method for direct search [17], and genetic algorithms [24], among many others discussed in [2, 11, 21, 22, 33, 34]. Furthermore, recent developments in DFO increasingly emphasize the use of surrogate and quadratic models to improve the quality of trial points and better exploit previously evaluated information. For instance, modern approaches combine quadratic interpolation with structured trust-region frameworks or subspace techniques to address high-dimensional or partially separable problems [18, 36]. Other works integrate adaptive sampling and machine learning models to improve surrogate accuracy and search efficiency [29, 32]. Recent model-and-search frameworks explicitly combine direct search strategies with quadratic model construction to enhance local convergence properties while maintaining robustness in blackbox settings [37]. These developments highlight the importance of optimization frameworks that can naturally incorporate model-based and heuristic search mechanisms without degrading the quality of the proposed trial points. More generally, related direct search convergence mechanisms based on stepsize or radius updates are surveyed in [27].

1.2 Contributions

This work introduces ADS-PB, a PB extension of the ADS framework, designed to preserve the flexibility of ADS while allowing infeasible evaluations to be incorporated throughout the optimization process. The proposed framework is intended for blackbox settings where constraint violations carry meaningful information and can be exploited to guide the search, rather than being discarded entirely.

A central aspect of ADS-PB is that it removes two structural features that often limit the effective use of sophisticated search mechanisms in direct search methods, namely the reliance on a mesh and the use of sufficient decrease conditions. By lifting these restrictions, the framework can directly exploit high-quality trial points generated by search heuristics, surrogate models, or hybrid strategies without altering them through mesh projection or rejecting them because of restrictive acceptance requirements. This makes ADS-PB particularly well suited to modern model-assisted DFO.

Beyond the algorithmic framework itself, this work also establishes a convergence analysis under standard assumptions of DFO, providing theoretical support for the proposed approach. Its practical relevance is further illustrated through an implementation within the `NOMAD` software [14] and through computational experiments on constrained analytical and blackbox test problems, where the behavior of ADS-PB is compared with that of existing DFO methods.

1.3 Outline

Section 2 recalls the key notions of the PB framework and fixes the terminology used throughout the paper. Section 3 presents the ADS-PB algorithmic framework. The convergence analysis is given in Section 3.4, and computational results, together with implementation aspects in `NOMAD`, are reported in Section 4. Concluding remarks are provided in Section 5.

2 The progressive barrier approach

A simple way to handle relaxable and quantifiable constraints is the two-phase extreme barrier approach: The first phase ignores the objective function and attempts to find a feasible point by minimizing a measure of constraint violation. The second phase focuses on reducing the objective function over the feasible set. This approach, although straightforward and simple to implement, might discard useful information from objective function evaluations at infeasible points. The PB approach [9] is designed to unify these two phases by allowing the algorithm to improve both feasibility and the objective while progressively restricting the admissible level of constraint violation.

2.1 Constraint violation function and incumbent solutions

The PB is a framework that may be combined with different direct search mechanisms: it can be coupled with a mesh structure, as in [9], but it also admits alternative realizations, for instance, merit-function-based approaches that balance objective decrease and feasibility improvement [6]. Constraints violations are aggregated using the *constraints violation function* $h : \mathbb{R}^n \to \overline{\mathbb{R}}^+$ defined by

$$h(x) \;=\; \begin{cases} \displaystyle\sum_{j \in J} \max(0, c_j(x))^2, & \text{if } x \in X, \\ +\infty, & \text{otherwise.} \end{cases}$$

With this notation, the feasible set may be written as $\Omega = \{x \in X :\ h(x) = 0\}$.

The main idea behind the PB approach is to use a *barrier threshold* $h^k_{\max}$ on the constraint violation function at each iteration k. The barrier threshold is initialized to $+\infty$. As the algorithm is deployed, when a candidate solution has a constraint violation function value that exceeds the current threshold, it is considered inadmissible for replacing one of the incumbents. The threshold is then progressively decreased, making the algorithm increasingly selective with respect to feasibility. Candidate solutions are compared against the current incumbents, which may be updated between successive iterations. Such comparisons are based either solely on the objective function value when the candidate is feasible or on both the objective function and constraint violation function values when the candidate is infeasible, as follows:

- A feasible point $x \in \Omega$ is preferred over $y \in \Omega$ (denoted by $x \prec_{\text{feas}} y$), if it improves the value of f (i.e., $f(x) < f(y)$).
- An infeasible point $x \in X \setminus \Omega$ is preferred over $y \in X \setminus \Omega$ (denoted by $x \prec_{\text{inf}} y$) if it improves at least one of f or h without worsening the other, i.e., either $f(x) < f(y)$ and $h(x) \leq h(y)$, or, $f(x) \leq f(y)$ and $h(x) < h(y)$.

In both situations, if $x \prec_{\text{feas}} y$ or $x \prec_{\text{inf}} y$, then x is said to *dominate* y, the notation is abbreviated by $x \prec y$.

The PB approach stores and orders the list of all trial points that were evaluated. Let $\mathcal{C}^k$, referred to as the *cache*, denote the set of all trial points where the objective and constraint functions were evaluated up to the start of iteration $k \in \mathbb{N}$. Any point in $\mathcal{C}^k$ is called a *visited point*. The algorithm maintains up to two types of incumbent solutions. The set of *feasible incumbents* is defined as

$$F^k \ := \ \underset{x \in \mathcal{C}^k}{\operatorname{argmin}} \{f(x) \ : \ h(x) = 0\}. \tag{2}$$

This set is empty when the cache $\mathcal{C}^k$ does not contain any feasible point. The second set of incumbents relies on the barrier threshold $h^k_{\max}$ and on the set of infeasible non-dominated points, i.e.,

$$U^k := \left\{ x \in \mathcal{C}^k \setminus \Omega \ : \ \text{there exists no } y \in \mathcal{C}^k \setminus \Omega \text{ such that } \ y \prec_{\text{inf}} x \right\}.$$

The set of *infeasible incumbents* is defined as

$$I^k \ := \ \underset{x \in U^k}{\operatorname{argmin}} \{f(x) \ : \ 0 < h(x) \leq h^k_{\max}\}. \tag{3}$$

With these definitions, F^k contains the feasible points visited so far with the least value of f, and I^k contains the infeasible non-dominated points visited so far (whose constraint violation function value is less than the barrier threshold) that have the least value of f.

These two sets of incumbents ensure that the algorithm reports a meaningful incumbent: either a best feasible point when feasibility has been reached, or a best infeasible point when it has not. To summarize the information carried by these sets, the framework associates up to two representative incumbent points at iteration k. If $F^k \neq \emptyset$, *a feasible incumbent* $x^k_{\mathrm{F}} \in F^k$ is selected, and its value is denoted by f^k_{F}, i.e.,

$$x^k_{\mathrm{F}} \in F^k \quad \text{and} \quad f^k_{\mathrm{F}} \ := \ f\big(x^k_{\mathrm{F}}\big)\,. \tag{4}$$

If $I^k \neq \emptyset$ is selected, an *infeasible incumbent* $x^k_{\mathrm{I}} \in I^k$ is chosen, together with its pair of values $(f^k_{\mathrm{I}}, h^k_{\mathrm{I}})$, i.e.,

$$x^k_{\mathrm{I}} \in I^k \quad \text{and} \quad (f^k_{\mathrm{I}}, h^k_{\mathrm{I}}) \ := \ \big(f(x^k_{\mathrm{I}}),\, h(x^k_{\mathrm{I}})\big). \tag{5}$$

In the case where F^k or I^k is not a singleton, the incumbents $x^k_{\mathrm{F}} \in F^k$ and $x^k_{\mathrm{I}} \in I^k$ may be chosen arbitrarily. These incumbents will serve as reference points to determine the outcome of iteration k: a new trial point at which f is evaluated is compared against x^k_{F} if it is feasible and against x^k_{I} if it is infeasible.

2.2 Dominating, improving and unsuccessful iterations

During iteration k, trial points may improve the current incumbents in different ways. A first possibility is through a *dominating* point. This includes the case of a feasible trial point whose objective function value is strictly less than f_{F}^k, when $F^k \neq \emptyset$, or any feasible trial point when $F^k = \emptyset$. Similarly, an infeasible trial point is considered dominating if it dominates x_{I}^k when $I^k \neq \emptyset$, or if it is the first infeasible point found when $I^k = \emptyset$. In all these cases, the iteration is declared *dominating*.

All iterations do not produce dominating points, especially before feasibility is reached. In that case, the PB still allows the algorithm to make progress by focusing on feasibility: in the situations where iteration k is not dominating , the algorithm may accept an *improving* point, that is, an infeasible visited point with a strictly smaller constraint violation function value than that of the current infeasible incumbent: an *improving point* $x \in X$ satisfies the following inequalities:

$$0 \ < \ h(x) \ < h_{\mathrm{I}}^k.$$

Any dominating or improving point is called a *successful point*. Finally, an iteration that is neither dominating nor improving is said to be *unsuccessful*.

Figure 1 illustrates the three iteration types in the (f, h)-plane. The feasible incumbent x_{F}^k lies on the vertical axis $h = 0$, while the infeasible incumbent x_{I}^k has coordinates $(f_{\mathrm{I}}^k, h_{\mathrm{I}}^k)$ with $h_{\mathrm{I}}^k > 0$. The dominance relation partitions the plane into three regions.

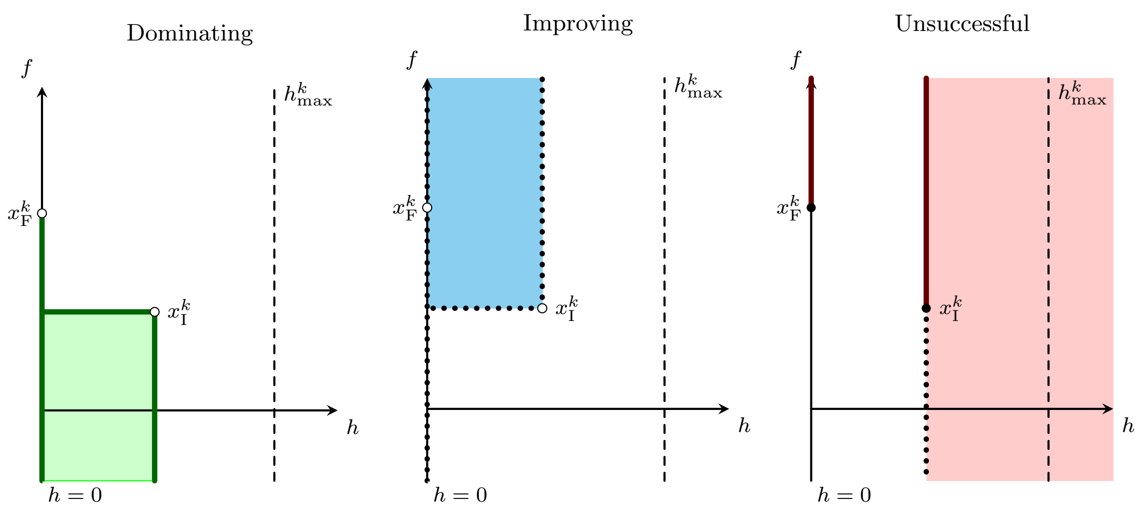


Figure 1: Partition of the (h, f)-plane induced by the infeasible incumbent x_{I}^k, the feasible incumbent x_{F}^k, and the barrier threshold $h_{\max}^k$. Solid bold lines and filled points are included in the shaded region, whereas dotted bold lines are excluded.

The *dominating* region consists of points that dominate the current reference (either feasible points on $h = 0$ with $f < f_{\mathrm{F}}^k$, highlighted in green on the leftmost figure, or infeasible points with $h \leq h_{\mathrm{I}}^k$ and $f \leq f_{\mathrm{I}}^k$ with at least one strict inequality). The *improving* region corresponds to infeasible points that strictly decrease the violation, $0 < h < h_{\mathrm{I}}^k$, even if f is not improved. It is illustrated in blue in the central figure. Finally, points that fall outside these two regions are *unsuccessful* for iteration k. The region is delimited in red in the rightmost figure. The barrier threshold $h_{\max}^k$ appears as a vertical line: it indicates the current admissible range of constraint violation for selecting the infeasible incumbents, and it helps visualize how the method becomes more and more restrictive as it progresses.

The barrier threshold parameter $h_{\max}^k$ monotonically evolves depending on whether the iteration is dominating, improving, or unsuccessful. The update procedure for $h_{\max}^k$ satisfies the following rules:

$$h_{\max}^{k+1} \ = \ \begin{cases} \max\limits_{x \in \mathcal{C}^{k+1}} \{h(x) : 0 < h(x) < h_{\max}^k\} & \text{if the iteration is improving,} \\ h_{\mathrm{I}}^{k+1} & \text{otherwise.} \end{cases} \tag{6}$$

The parameter $h_{\max}^k$ gradually decreases toward zero and determines the tolerance allowed for constraint violations. When the iteration is improving, the idea is to push the infeasible incumbent values toward the feasible region even when iteration k does not generate a point that dominates the current incumbents. After a dominating or unsuccessful iteration, this parameter is set equal to h_{I}^k. In an improving iteration, the existence of a visited point t such that $0 < h(t) < h(x_{\mathrm{I}}^k)$ implies that $U^k \setminus \{x_{\mathrm{I}}^k\}$ is nonempty. It is therefore possible to reduce the threshold $h_{\max}^k$ while keeping at least one admissible infeasible point. The chosen strategy is to set $h_{\max}^k$ to the largest value of $h(x)$ that is strictly smaller than the previous threshold and such that x is an improving point. Setting $h_{\max}^k = 0$ would reduce the problem to the minimization of the *extreme barrier function* f_Ω, where $f_\Omega = f$ on Ω and $f_\Omega = +\infty$ elsewhere.

The general PB framework is summarized in Algorithm 1. In the step 2, the set $\mathbb{Y}^k$ of points generated at iteration k is arbitrary.

Algorithm 1 **The Progressive Barrier (PB) framework.**

▷ **Step 0. Initialization:**
Initialize $\mathcal{C}^0 \leftarrow \{x^0\}$, set $h_{\max}^0 \leftarrow +\infty$ and $k \leftarrow 0$

▷ **Step 1. Incumbents definitions:**
Set x_{F}^k and x_{I}^k according to (4) and (5), respectively.

▷ **Step 2. Function evaluations:**
Generate $\mathbb{Y}^k$, evaluate $(f(y), h(y))$ for all $y \in \mathbb{Y}^k$, and set $\mathcal{C}^{k+1} \leftarrow \mathcal{C}^k \cup \mathbb{Y}^k$

▷ **Step 3. Update step:**

$$\text{Declare iteration } k \begin{cases} \textit{dominating} & \text{if there exists a dominating point } y \in \mathcal{C}^{k+1} \\ \textit{improving} & \text{else, if there exists an improving point } y \in \mathcal{C}^{k+1} \\ \textit{unsuccessful} & \text{otherwise} \end{cases}$$

Update $h_{\max}^{k+1}$ according to (6)

▷ **Step 4. Termination:**
if no termination test is triggered **then** $k \leftarrow k+1$ and return to **Step 1**

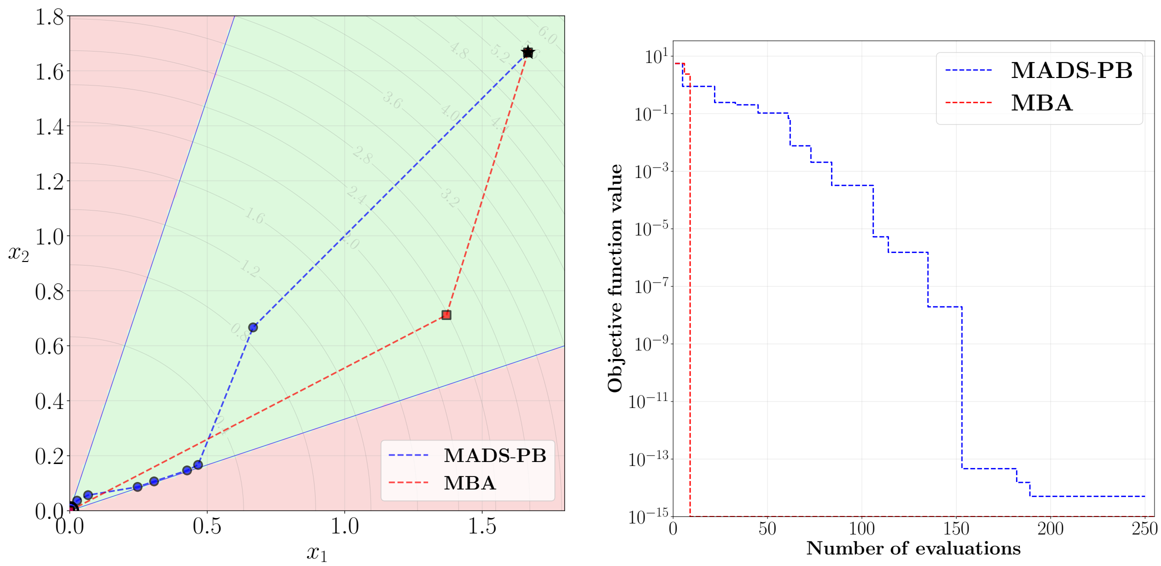
2.3 A motivational toy example

While the PB provides a general framework to handle constraints, it does not, by itself, ensure convergence; such guaranties are inherited from the optimization algorithm in which it is embedded. With the Mesh Adaptive Direct Search (MADS) algorithm [9], these guaranties rely on the use of a mesh that discretizes the search space and governs the generation of trial points. This dependence on the mesh, although central to the convergence analysis, can also introduce limitations. The MADS candidate points are generated on a discretization of the space, controlled by a mesh size parameter. When the algorithm proposes a point that is not on the current mesh, the point is projected onto a nearby mesh point before evaluation. This mechanism is central in MADS, but it can impede the benefit of accurate model-based information: even if a local model suggests a trial point close to a local minimizer x^*, the mesh projection may move it away, and the algorithm would need to undergo many mesh refinements before evaluating f on a trial point close to x^*.

To illustrate this last observation, consider the smooth convex problem in $\mathbb{R}^2$

$$\min_{(x_1,x_2)\in\mathbb{R}^2} f(x_1,x_2) = x_1^2 + x_2^2 \quad \text{s.t.} \quad x_1 - 3x_2 \le 0 \text{ and } x_2 - 3x_1 \le 0,$$

whose unique optimal solution is $x^\star = (0,0)$. Figure 2 illustrates the obtained results of two DFO algorithms using the starting point $(5/3, 5/3)$. In Figure 2a, the feasible domain Ω is represented in green. After a small number of function evaluations, a quadratic model can be built to locate exactly the optimal solution $x^\star$. With MADS-PB [9] (i.e. MADS with the PB), all trial points need to be projected onto a mesh, which forces additional evaluations until the mesh becomes fine enough to locate $x^\star$ with sufficient precision. The left part of the figure shows the feasible incumbent solutions produced by MADS-PB and by a model-based algorithm (MBA), which does not project trial points onto a mesh. In this setting, the MBA builds local quadratic models of the objective and constraint functions from previously visited points, and proposes trial points by solving the resulting surrogate problem. In this example, as shown in Figure 2b, MADS-PB fails to evaluate f on the exact optimal point, while the MBA reaches $x^\star$ in fewer than ten evaluations.



(a) Trajectory plots. **(b) Convergence plots.**

Figure 2: Trajectory and convergence plots for the MADS-PB method and a model-based algorithm (MBA).

3 Adaptive direct search with a progressive barrier

This section describes the main contributions of this work: the ADS-PB algorithm, based on the general PB framework described in Algorithm 1. The ADS-PB algorithm extends ADS [10] to constrained blackbox optimization problems with quantifiable and relaxable constraints by incorporating the PB mechanism of MADS-PB [9]. The main purpose is to retain the PB framework while replacing the mesh-based structure of MADS-PB by the mesh-free structure of ADS. In particular, the step 2 is decomposed into the *Search* and *Poll steps*. In addition, a fourth iteration status is introduced in the *Update step*: iteration k can be declared *reframing*. The overall structure of the method is summarized in Algorithm 2.

3.1 The unrestricted search step

The *search* step at iteration k of ADS-PB produces a finite, possibly empty set of trial points in X denoted by $\mathbb{S}^k$. The generation of these points is flexible and may exploit information from previously visited points: let $\mathbb{V}^k_{\text{succ}}$ denote the set of successful visited points up to the start of iteration k. The first set $\mathbb{V}^0_{\text{succ}}$ is initialized to $\{x^0\}$ where $x^0 \in X$.

Unlike MADS-PB, the search set $\mathbb{S}^k$ is not restricted to a mesh but to a punctured space consisting of $\mathbb{R}^n$ deprived of balls around all points of $\mathbb{V}^k_{\text{succ}}$. The radii of the balls are determined by the *exclusion size parameter* $\delta^k > 0$, which is updated at the end of each iteration. This parameter plays a role similar to that of the MADS mesh size parameter.

Definition 1. The *punctured space* $\mathring{\mathbb{R}}^n_k$ at iteration k of ADS-PB is the set of points in $\mathbb{R}^n$ that are not within δ^k of $\mathbb{V}^k_{\text{succ}}$, i.e.,

$$\mathring{\mathbb{R}}^n_k \;:=\; \left\{x \in \mathbb{R}^n : \|x-y\| \geq \delta^k \text{ for all } y \in \mathbb{V}^k_{\text{succ}}\right\} \;. \tag{7}$$

With ADS-PB, there are three possible outcomes of the search step:

1. A dominating trial point $t_{\text{search}} \in \mathbb{S}^k$ inside $\mathring{\mathbb{R}}^n_k$ is found: As soon as this occurs, in *opportunistic* settings the search is interrupted, the iteration is declared *dominating*, ADS-PB skips the poll step, and pursues to the update step. Note that during the poll step the notion of dominating uses p^k_{F} and p^k_{I} as reference instead of x^k_{F} and x^k_{I}.
2. A dominating trial point $t_{\text{search}} \in \mathbb{S}^k$ outside $\mathring{\mathbb{R}}^n_k$ is found: As soon as this occurs, the search is interrupted and ADS-PB performs the poll step around t_{search} (described in the next subsection). The iteration will be declared *dominating* if the poll step identifies a dominating point, and *reframing* if the poll step fails to do so.
3. All trial points of $\mathbb{S}^k$ are evaluated, and none of them is dominating.
   ADS-PB performs the poll step around the incumbent(s) solution(s) (described in the next subsection). The poll step will determine if the iteration is declared *dominating*, *improving* or *unsuccessful*.

The set of all points on which f is evaluated by the algorithm at the search step of iteration k is denoted $\mathbb{S}^k_{\text{eval}}$. This set might be a strict subset of $\mathbb{S}^k$ when the search step is opportunistically interrupted.

3.2 The poll step confined to the punctured space

The *poll* step is a local exploration around the best feasible and infeasible solutions found so far. The algorithm defines the *poll centers* as follows:

Definition 2 (Poll centers)**.** The feasible and infeasible poll centers are defined as

$$p^k_{\text{F}} = \begin{cases} t_{\text{search}} & \text{if } t_{\text{search}} \in \Omega \text{ and dominating,} \\ x^k_{\text{F}}, & \text{otherwise if defined.} \end{cases} \quad p^k_{\text{I}} = \begin{cases} t_{\text{search}}, & \text{if } t_{\text{search}} \notin \Omega \text{ and dominating,} \\ x^k_{\text{I}}, & \text{otherwise if defined.} \end{cases}$$

In other words, at the beginning of the poll step of iteration k, p^k_{F} denotes the best feasible point identified so far, while p^k_{I} denotes the best infeasible point identified so far among those satisfying $h(x) \le h^k_{\max}$. Moreover, there are either one or two poll centers since $F^k \cup I^k$ is nonempty (since $x^0 \in F^0 \cup I^0$).

Once that the poll centers are determined, the poll set $\mathbb{P}^k$ is built from the sets $\mathbb{D}^k_{\text{I}}$ and $\mathbb{D}^k_{\text{F}}$ of normalized *poll directions*. These directions, i.e., unit vectors defining the directions along which trial points are generated, are selected to form a positive spanning set [23], ensuring adequate coverage of the local region around p^k_{F} and p^k_{I}. The *tentative poll points* are then obtained by moving from the poll centers along these directions, scaled by the *frame size parameter* $\Delta^k \in \mathbb{R}^*_+$, introduced in [8], since this parameter plays the same role as in MADS-PB. The poll set is defined as

$$\mathbb{P}^k := \begin{cases} \{p^k_{\text{F}} + \Delta^k v : v \in \mathbb{D}^k_{\text{F}}\} & \text{if } I^k = \emptyset, \\ \{p^k_{\text{I}} + \Delta^k v : v \in \mathbb{D}^k_{\text{I}}\} & \text{if } F^k = \emptyset, \\ \{p^k_{\text{F}} + \Delta^k v : v \in \mathbb{D}^k_{\text{F}}\} \cup \{p^k_{\text{I}} + \Delta^k v : v \in \mathbb{D}^k_{\text{I}}\} & \text{otherwise.} \end{cases}$$

As with ADS, the parameter Δ^k determines the distance between the trial points in $\mathbb{P}^k$ and the poll centers p^k_{I} and p^k_{F}. Because the poll directions are normalized, every poll point lies exactly at distance Δ^k from its associated poll center. When Δ^k decreases, the tentative poll points move closer to p^k_{I} and p^k_{F}, allowing finer local improvements. All tentative poll points in $\mathbb{P}^k$ that fall outside the punctured space $\mathring{\mathbb{R}}^n_k$ are discarded without evaluation, as such points are within distance δ^k of some element of $\mathbb{V}^k_{\text{succ}}$.

There are two possible outcomes of the poll step:

1. The poll step successfully produces a trial point $t_{\text{poll}} \in \mathring{\mathbb{R}}^n_k \cap \mathbb{P}^k$ that is dominating. In an opportunistic setting, the poll step is immediately interrupted.

2. After evaluating all points of $\mathring{\mathbb{R}}^n_k \cap \mathbb{P}^k$, no dominating trial point is found.

In the situation where the search step of iteration k visits a feasible or infeasible dominating point $t_{\text{search}} \in \mathbb{S}^k_{\text{eval}}$ outside of $\mathring{\mathbb{R}}^n_k$, then either $p^k_{\text{F}} = t_{\text{search}}$ or $p^k_{\text{I}} = t_{\text{search}}$. Two situations may occur. If the subsequent poll step fails to identify a dominating point, then the iteration is declared *reframing* and t_{search} becomes one of the next incumbents. Alternately, if the poll step produces a dominating point t_{poll}, the iteration is declared *dominating* and one incumbent is updated to t_{poll}. In both cases, at least one incumbent is updated at iteration k.

Denote by $\mathbb{P}^k_{\text{eval}}$ the set of tentative poll points at which f is evaluated during the poll step. Once this step is completed, an update step is performed.

3.3 Decision and parameters update

The *update* step is the final stage of iteration k. This step consists in analyzing the visited points during iteration k and the points from $\mathbb{C}^k$ in order to update all the parameters of ADS-PB. When either the search or the poll step produced a dominating point in $\mathring{\mathbb{R}}^n_k$, then iteration k is declared *dominating*. Otherwise, before declaring the iteration unsuccessful or reframing, the algorithm checks whether iteration k can be declared *improving*.

At iteration k, $\mathbb{C}^k \cup \mathbb{S}^k_{\text{eval}} \cup \mathbb{P}^k_{\text{eval}}$ denotes the set gathering all points visited before iteration k together with the trial points visited during the search and poll steps of iteration k. Iteration k is said to be *improving* if there exists a non dominating point $t_{\text{update}} \in \mathcal{C}^k \cup \mathbb{S}^k_{\text{eval}} \cup \mathbb{P}^k_{\text{eval}}$ such that

$$t_{\text{update}} \in \mathring{\mathbb{R}}^n_k \quad \text{and} \quad 0 < h(t_{\text{update}}) < h\left(x^k_{\text{I}}\right). \tag{8}$$

By construction, the point t_{update} must be an infeasible point that is strictly better than the current infeasible incumbent x^k_{I} in terms of constraint violation. Such a point is called an *improving point*.

Finally, if an iteration is neither dominating nor improving, then it is declared *reframing* whenever the search step has produced a dominating point outside $\mathring{\mathbb{R}}^n_k$, and *unsuccessful* otherwise. This terminology (i.e., a *reframing* iteration) is motivated by the fact that, even if no dominating point is found within the punctured space, a dominating search point outside $\mathring{\mathbb{R}}^n_k$ changes the incumbents. As a result, the poll centers as well as the corresponding frames are repositioned. The update procedure for the barrier threshold parameter $h^k_{\max}$ follows the PB rule of [9] and is recalled in Section 2, see (6). In particular, the threshold is aligned with the current infeasible incumbent after a dominating, reframing or unsuccessful iteration, while an improving iteration decreases the threshold to the largest violation level strictly below the previous one among the infeasible visited points.

The update procedure for the frame size and exclusion parameters is defined as follows:

$$(\Delta^{k+1}, \delta^{k+1}) = \begin{cases} \texttt{increase}(\Delta^k, \delta^k) & \text{if the iteration } k \text{ is dominating,} \\ (\Delta^k, \delta^k) & \text{if the iteration } k \text{ is improving,} \\ \texttt{decrease}(\Delta^k, \delta^k) & \text{otherwise.} \end{cases} \tag{9}$$

The `decrease` and `increase` rules are defined as in [10, Equations (2) and (3)]. They allow that if $\lim_{k\in\mathbb{N}} \delta^k = 0$, then $\lim_{k\in\mathbb{N}} \dfrac{\delta^k}{\Delta^k} = 0$ and $\lim_{k\in\mathbb{N}} \Delta^k = 0$. The set $\mathbb{V}^k_{\text{succ}}$ is then updated as follows.

$$\mathbb{V}^{k+1}_{\text{succ}} = \begin{cases} \mathbb{V}^k_{\text{succ}} \cup \{t_{\text{search}}\} & \text{if } t_{\text{search}} \in \mathbb{S}^k_{\text{eval}} \cap \mathring{\mathbb{R}}^n_k \text{ is dominating} \\ \mathbb{V}^k_{\text{succ}} \cup \{t_{\text{poll}}\} & \text{if } t_{\text{poll}} \in \mathbb{P}^k_{\text{eval}} \text{ is dominating,} \\ \mathbb{V}^k_{\text{succ}} \cup \{t_{\text{update}}\} & \text{if } t_{\text{update}} \in \mathbb{C}^k \cup \mathbb{S}^k_{\text{eval}} \cup \mathbb{P}^k_{\text{eval}} \cap \mathring{\mathbb{R}}^n_k \text{ is improving,} \\ \mathbb{V}^k_{\text{succ}} & \text{otherwise.} \end{cases} \tag{10}$$

Note that the "otherwise" cases in (9) and (10) actually encompass two distinct situations, namely reframing and unsuccessful iterations. Although both cases lead to the same updates of the parameters (Δ^k, δ^k) and the set $\mathbb{V}^k_{\text{succ}}$, the outcome of the iteration is fundamentally different. In the case of a reframing iteration, an incumbent solution is guaranteed to be modified, reflecting a meaningful improvement. In contrast, during an unsuccessful iteration, no incumbent is updated and no progress is made in terms of dominance or improvement.

In Algorithm 2, the temporary set T^k_{succ} is introduced to ensure compact notation. The set of successfully visited points $\mathbb{V}^k_{\text{succ}}$ replaces the set $\mathbb{V}^k$ introduced in [10] and is used to construct the punctured space $\mathring{\mathbb{R}}^n_k$. At the end of each iteration, the set $\mathbb{V}^k_{\text{succ}}$ may be updated by adding at most one new point. A point is added to $\mathbb{V}^k_{\text{succ}}$ only when iteration k is declared improving or dominating. Moreover, whenever iteration k is improving or dominating, the point added to $\mathbb{V}^k_{\text{succ}}$ is guaranteed to be at least δ^k units away from all previously stored points in $\mathbb{V}^k_{\text{succ}}$.

Algorithm 2 **Adaptive Direct Search with Progressive Barrier (ADS-PB).**

▷ **Step 0. Initialization:**
- $x^0 \in X$, $\Delta^0 \in \mathbb{R}^*_+$, $\delta^0 \in \mathbb{R}^*_+$ such that $\delta^0 \leq \Delta^0$
- $h^0_{\max} \leftarrow +\infty$, $k \leftarrow 0$ ▷ initial parameters
- $\mathcal{C}^0 \leftarrow \{x^0\}$ ▷ initial set of cache points
- $\mathbb{V}^0_{\text{succ}} \leftarrow \{x^0\}$ ▷ initial set of successful points
- $\mathring{\mathbb{R}}^n_0 \leftarrow \{x \in \mathbb{R}^n : \|x - x^0\| \geq \delta^0\}$ ▷ initial punctured space

▷ **Step 1. Incumbents definition:**
- Set the incumbents x^k_{F} and x^k_{I} (see Section 2.1) and $T^k_{\text{succ}} \leftarrow \emptyset$

▷ **Step 2. Function evaluations:**
- ▷ **Step 2.1. Search step:**
  - Define a finite search set $\mathbb{S}^k \subset X$
  - **if** t_{search} is *dominating* for some $t_{\text{search}} \in \mathbb{S}^k_{\text{eval}}$ **then**
    - **if** $t_{\text{search}} \in \mathring{\mathbb{R}}^n_k$ **then**
      - Set $T^k_{\text{succ}} \leftarrow \{t_{\text{search}}\}$ and go to **Step 3**
- ▷ **Step 2.2. Poll step:**
  - Define the poll set $\mathbb{P}^k = \mathbb{P}^k_{\text{F}} \cup \mathbb{P}^k_{\text{I}}$ around p^k_{F} and p^k_{I} (see Definition 2)
  - **if** t_{poll} is *dominating* for some $t_{\text{poll}} \in \mathbb{P}^k_{\text{eval}} \cap \mathring{\mathbb{R}}^n_k$ **then**
    - Set $T^k_{\text{succ}} \leftarrow \{t_{\text{poll}}\}$

▷ **Step 3. Update step:**
- Set $\mathcal{C}^{k+1} \leftarrow \mathcal{C}^k \cup \mathbb{S}^k_{\text{eval}} \cup \mathbb{P}^k_{\text{eval}}$

$$\text{Declare iteration } k \begin{cases} \textit{dominating} & \textbf{if } T^k_{\text{succ}} \neq \emptyset \\ \textit{reframing} & \textbf{else, if } \text{the search step produced a dominating point} \\ \textit{improving} & \textbf{else, if } \text{there is } t_{\text{update}} \in \mathcal{C}^{k+1} \cap \mathring{\mathbb{R}}^n_k \text{ an improving point} \\ \textit{unsuccessful} & \textbf{otherwise} \end{cases}$$

- **if** iteration k is *improving* **then** set $T^k_{\text{succ}} \leftarrow \{t_{\text{update}}\}$
- Update $\mathbb{V}^{k+1}_{\text{succ}} \leftarrow \mathbb{V}^k_{\text{succ}} \cup T^k_{\text{succ}}$, and $\mathring{\mathbb{R}}^n_{k+1}$ according to (7)
- Update $h^{k+1}_{\max}$ and $(\Delta^{k+1}, \delta^{k+1})$ according to (6) and (9)

▷ **Step 4. Termination:**
- **if** no termination test is triggered **then** $k \leftarrow k + 1$ and go to **Step 1**

This preserves the foundational principles of directional direct search while removing the limitations imposed by mesh structures in MADS-PB. The set of successfully visited points $\mathbb{V}^k_{\text{succ}}$ replaces the set $\mathbb{V}^k$ introduced in [10] and is used to construct the punctured space $\mathring{\mathbb{R}}^n_k$. At the end of each iteration, the set $\mathbb{V}^k_{\text{succ}}$ may be updated by adding up to one new point. A point is added to $\mathbb{V}^k_{\text{succ}}$ only when iteration k is declared improving or dominating.

There is three main differences between ADS-PB and MADS-PB. First, the search step of ADS-PB is completely free in X: any trial point in the blackbox domain may be evaluated, without projection onto a mesh. Second, before evaluating poll points, ADS-PB checks whether they belong to the punctured space $\mathring{\mathbb{R}}^n_k$, so as to avoid evaluating points that are too close to previously successful points.

Third, this mechanism introduces a new possible qualification of the iteration, called a reframing iteration, which occurs when the search step finds a successful point outside the punctured space and then the poll fails to find a dominating point.

3.4 Convergence analysis

This section presents the convergence analysis of ADS-PB. As is common in direct search [7], the analysis first studies the behavior of δ^k and Δ^k as k goes to infinity, and then proposes stationary results.

Assumption 1. All the points at which the objective and constraint functions are evaluated belong to a compact set L.

Under Assumption 1, the exclusion radius parameter δ^k and the frame size parameter Δ^k converge to zero as the number of iterations approaches infinity.

Theorem 3.1. Let Assumption 1 hold. The exclusion radius parameter sequence $\{\delta^k\}_{k\in\mathbb{N}}$ and the frame size sequence $\{\Delta^k\}_{k\in\mathbb{N}}$ produced by an instance of ADS-PB satisfy

$$\lim_{k\to\infty} \delta^k \;=\; 0 \qquad \text{and} \qquad \lim_{k\to\infty} \Delta^k \;=\; 0.$$

Proof. Let $\varepsilon \in \mathbb{R}^*_+$ and define $\mathcal{K}_\varepsilon := \left\{k \in \mathbb{N} : \delta^k \geq \varepsilon\right\}$, a subset of iteration indices.

Suppose that $\mathcal{K}_\varepsilon$ contains infinitely many elements. Let $\mathcal{S}_\varepsilon \subseteq \mathcal{K}_\varepsilon$ denote the subset of indices of dominating and improving iterations k for which $\delta^k \geq \varepsilon$. Two scenarios can occur depending on the cardinality of $\mathcal{S}_\varepsilon$:

- $|\mathcal{S}_\varepsilon|$ is finite, in which case there exists $k_0 \in \mathbb{N}$ such that the iteration k is unsuccessful or reframing for all $k \geq k_0$. It follows that $\lim_{k\in\mathcal{K}_\varepsilon} \delta^k = 0$, contradicting $\delta^k \geq \varepsilon$, for all $k \in \mathcal{K}_\varepsilon$.
- $|\mathcal{S}_\varepsilon|$ is infinite. For any $k \in \mathcal{S}_\varepsilon$, let $t^k \in \{t_{\text{search}}, t_{\text{poll}}, t_{\text{update}}\}$ denote the improving or dominating point found during the iteration. Since the iteration is either dominating or improving, it follows that t^k belongs to $\mathring{\mathbb{R}}^n_k$ and

  $$t^k \;\in\; \left\{t \in X \;:\; \|t - t^j\| \geq \delta^k,\; j \in \mathcal{S}_\varepsilon \text{ and } j < k\right\}.$$

  Hence, any pair of indices $k_1 \neq k_2$ in $\mathcal{S}_\varepsilon$ satisfy the inequality

  $$\|t^{k_1} - t^{k_2}\| \;\geq\; \inf_{i\in\mathcal{S}_\varepsilon} \delta^i \;\geq\; \varepsilon\,. \tag{11}$$

  However, since $\{t^k\}_{k\in\mathcal{S}_\varepsilon}$ belongs to the compact set L (from Assumption 1), the Bolzano-Weierstrass theorem ensures the existence of a convergent subsequence, contradicting (11).

Both scenarios lead to a contradiction, and therefore the set $\mathcal{K}_\varepsilon$ is finite. Finally, the discussion following (9) ensures that $\{\delta^k\}_{k\in\mathbb{N}} \to 0$ implies that $\{\Delta^k\}_{k\in\mathbb{N}} \to 0$. □

Theorem 3.1 is stronger than the corresponding result in the MADS-PB context [8, Proposition 3.1], as the limit is shown to be zero, instead of simply ensuring that the limit inferior is zero. This stronger limit result is similar to those derived in sufficient decrease contexts [26, Theorem 4.1] and [16, Lemma 3.2].

The ADS-PB algorithm generates a sequence of trial points from which one can extract one or two subsequences of poll centers: the feasible poll centers $\{p^k_{\mathrm{F}}\}_{k\in\mathbb{N}}$ and the infeasible poll centers $\{p^k_{\mathrm{I}}\}_{k\in\mathbb{N}}$. The following convergence result focuses on the subsequence of feasible poll centers.

Definition 3 (A refining subsequence)**.** [8, Definition 3.1] A convergent subsequence $\{p^k\}_{k\in\mathcal{K}}$, where $\mathcal{K}$ is an infinite subset of $\mathbb{N}$, of unsuccessful iterations is said to be a refining subsequence if $\lim_{k\in\mathcal{K}} \Delta^k = 0$. The limit $\hat{x}$ of a convergent refining subsequence is called a refined point. If $\hat{v} = \lim_{k\in\mathcal{L}} v^k$ exists for some $\mathcal{L} \subseteq \mathcal{K}$ with poll direction $v^k \in \mathbb{P}^k$ and if $p^k + \Delta^k v^k \in X$ for infinitely many $k \in \mathcal{L}$, then $\hat{v}$ is said to be a refining direction for $\hat{x}$.

The notion of a refining subsequence makes it possible to characterize the asymptotic behavior of the sequence of feasible incumbents along unsuccessful iterations as the frame size parameter tends to zero. For the next convergence results, one must guaranty that $p+t\hat{v}$ belongs to Ω when $p \in \Omega$ is close to $\hat{x}$, which is ensured if $\hat{v}$ is a hypertangent vector to Ω at $\hat{x}$.

Definition 4. [39] A vector $\hat{v} \in \mathbb{R}^n$ is said to be a hypertangent vector to the set $\Omega \subseteq \mathbb{R}^n$ at the point $\hat{x} \in \Omega$ if there exists a scalar $\epsilon > 0$ such that

$$p + tv \in \Omega \qquad \forall p \in \Omega \cap B_\epsilon(\hat{x}),\ v \in B_\epsilon(\hat{v}), \ \text{ and } \ 0 < t < \epsilon.$$

The hypertangent cone to Ω at $\hat{x}$, denoted by $T^H_\Omega(\hat{x})$, is the set of all hypertangent vectors to Ω at $\hat{x}$. In this case, for $\hat{x} \in \Omega$ and $\hat{v} \in T^{\mathrm{H}}_\Omega(\hat{x})$, if f is Lipschitz continuous near $\hat{x}$, the Clarke Jahn generalized derivative, as defined in [19] and extended to constrained settings in [31], is given by

$$f^\circ(\hat{x};\hat{v}) := \limsup_{\substack{p\to\hat{x},\ t\searrow 0,\\ p+t\hat{v}\in\Omega,\ p\in\Omega}} \frac{f(p+t\hat{v}) - f(p)}{t}.$$

The next theorem establishes that any feasible refined point is Clarke stationary with respect to every refining direction in $T^{\mathrm{H}}_\Omega(\hat{x}_{\mathrm{F}})$.

Theorem 3.2. Let Assumption 1 hold. If $\{p^k_{\mathrm{F}}\}_{k\in\mathcal{K}}$ is a refining subsequence converging to $\hat{x}_{\mathrm{F}} \in \Omega$, near which f is Lipschitz continuous, then for all refining directions $\hat{v} \in T^{\mathrm{H}}_\Omega(\hat{x}_{\mathrm{F}})$,

$$f^\circ(\hat{x}_{\mathrm{F}};\hat{v}) \geq 0.$$

Proof. Let $\{p^k_{\mathrm{F}}\}_{k\in\mathcal{K}}$, be a refining subsequence composed of feasible poll centers, with refined point $\hat{x}_{\mathrm{F}} \in \Omega$, and suppose that $\hat{v} = \lim_{k\in\mathcal{L}} v^k$ is a refining direction.

The definition of a refining sequence ensures that $\lim_{k\in\mathcal{L}} \Delta^k = 0$ and $p^k_{\mathrm{F}} = \hat{x}_{\mathrm{F}}$ for all $k \in \mathcal{L}$.

Now, consider $t^k_{\mathrm{F}} = p^k_{\mathrm{F}} + \Delta^k v^k$, the poll point at iteration $k \in \mathcal{L}$ associated to the direction v^k. Two cases need to be analyzed. First, if t^k_{F} does not belongs to the punctured space $\mathring{\mathbb{R}}^n_k$, then the trial point t^k_{F} is rejected as there exists a previously visited point $y^k \in \mathcal{C}^k \subseteq \mathcal{C}^{k+1}$, such that $\|p^k_{\mathrm{F}} + \Delta^k v^k - y^k\| < \delta^k$. The direction $w_k = \frac{1}{\delta^k}(y^k - t^k)$ satisfies $\|w^k\| \leq 1$ and

$$y^k \ = \ t^k_{\mathrm{F}} + \delta^k w^k \ = \ p^k_{\mathrm{F}} + \Delta^k v^k + \delta^k w^k \ = \ p^k_{\mathrm{F}} + \Delta^k \left(v^k + \tfrac{\delta^k}{\Delta^k} w^k\right) \ \in \ \mathcal{C}^{k+1}. \tag{12}$$

Second, if t^k_{F} belongs to the punctured space $\mathring{\mathbb{R}}^n_k$, then define $y^k = t^k_{\mathrm{F}}$ and set $w^k = 0$ so that (12) remains valid. In both cases, $y^k \in \mathcal{C}^{k+1}$ and $f(y^k) \geq f(p^k_{\mathrm{F}})$.

Now, Theorem 3.1 ensures that $\lim_{k\in\mathcal{L}} \delta^k = \lim_{k\in\mathcal{L}} \Delta^k = 0$, and so $\lim_{k\in\mathcal{L}} \dfrac{\delta^k}{\Delta^k} = 0$ which implies that

$$\lim_{k\in\mathcal{L}} v^k + \frac{\delta^k}{\Delta^k} w^k \ = \ \hat{v}\,.$$

Furthermore, since $\hat{v}$ is an hypertangent direction, there exists an infinite subset $\mathcal{W} \subseteq \mathcal{L}$, such that for all $k \in \mathcal{W}$,

$$y^k,\ p^k_{\mathrm{F}} \ \in \ \Omega \quad \text{ and } \quad f(y^k) \ \geq \ f(p^k_{\mathrm{F}}).$$

Since f is Lipschitz continuous near $\hat{x}_{\mathrm{F}}$, the Clarke derivative exists. To derive a lower bound, consider the following three sequences indexed by $k \in \mathcal{W}$

$$p_{\mathrm{F}}^k \to \hat{x}_{\mathrm{F}}, \qquad \Delta^k \searrow 0 \quad \text{and} \quad v^k + \frac{\delta^k}{\Delta^k} w^k \to \hat{v}$$

and by [8, Proposition 3.9], it follows that

$$\begin{aligned} f^\circ(\hat{x}_{\mathrm{F}}; \hat{v}) \; &= \limsup_{\substack{x \to \hat{x}_{\mathrm{F}},\ t \searrow 0,\\ x + t\hat{v} \in \Omega,\ x \in \Omega}} \frac{f(x + t\hat{v}) - f(x)}{t} = \limsup_{\substack{x \to \hat{x}_{\mathrm{F}},\ t \searrow 0,\\ x + tv \in \Omega,\ x \in \Omega\\ v \to \hat{v}}} \frac{f(x + tv) - f(x)}{t} \\ &\geq \limsup_{k \in \mathcal{W}} \frac{f\left(p_{\mathrm{F}}^k + \Delta^k \left(v^k + \dfrac{\delta^k}{\Delta^k} w^k\right)\right) - f(p_{\mathrm{F}}^k)}{\Delta^k} \\ &= \limsup_{k \in \mathcal{W}} \frac{f(y^k) - f(p_{\mathrm{F}}^k)}{\Delta^k} \;\geq\; 0. \end{aligned}$$

□

The next result focuses on the sequence of infeasible poll centers $\{p_{\mathrm{I}}^k\}_{k \in \mathbb{N}}$. There are three possible outcomes for this sequence: the first is that no infeasible point is ever found, in which case the sequence is not defined; the second is that there exists a refining infeasible subsequence converging to a refined point $\hat{x}_{\mathrm{I}}$ in the feasible region Ω; in this case, ADS-PB found a global minimizer of h; the last case is that there exists a refining infeasible subsequence converging to a refined point outside the feasible region Ω; in this case, a necessary condition for $\hat{x}_{\mathrm{I}}$ to be a local minimizer of h is given in the following theorem.

Theorem 3.3. Let Assumption 1 hold. If $\{p_{\mathrm{I}}^k\}_{k \in \mathcal{K}}$ is a refining subsequence of infeasible poll centers converging to $\hat{x}_{\mathrm{I}} \in X$ near which h is Lipschitz with $x_{\mathrm{I}}^k \in I^k$, then for all refining directions $\hat{v} \in T_X^{\mathrm{H}}(\hat{x}_{\mathrm{I}})$

$$h^\circ(\hat{x}_{\mathrm{I}}; \hat{v}) \geq 0.$$

Proof. Indeed, if $\hat{x}_{\mathrm{I}} \in \Omega$, then $h(\hat{x}_{\mathrm{I}}) = 0$. Since $h(x) \geq 0$ for all $x \in X$, it follows that $\hat{x}_{\mathrm{I}}$ is a global minimizer of h on X, and the result holds. Otherwise, if $\hat{x}_{\mathrm{I}} \in X \setminus \Omega$, then by continuity of h, there exists $\varepsilon > 0$ such that $h(x) > 0$ for all $x \in B_\varepsilon(\hat{x}_{\mathrm{I}})$. This ensures the existence of an infinite subset $\mathcal{L} \subseteq \mathcal{K}$ such that, for all $k \in \mathcal{L}$, the sequence y^k defined in (12) satisfies $y^k, p_{\mathrm{I}}^k \in X \setminus \Omega$ and $h(y^k) \geq h(p_{\mathrm{I}}^k)$.

The conclusion follows by applying the same reasoning as in the proof of Theorem 3.2, replacing f by h, F^k by I^k, and p_{F}^k by p_{I}^k. □

4 Computational results

This section evaluates the practical performance of ADS-PB on a collection of constrained optimization problems. The proposed framework is tested on both analytical test problems and constrained blackbox problems with quantifiable and relaxable constraints. The objective is to empirically assess the behavior of ADS-PB and to compare it with existing DFO methods.

The experiments are conducted using the `NOMAD` software [14], which provides a rich algorithmic environment for DFO. In particular, the computational tests reported in this section benefit from several search strategies available in the solver, including a quadratic model-based search [20], a Nelder-Mead search [15], and a speculative search strategy [8]. The proposed approach is evaluated against other algorithms implemented within `NOMAD` and against other DFO solvers. These include the Covariance

Matrix Adaptation Evolution Strategy (CMA-ES) [30], the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [24] and the mixed penalty-interior point method and direct search (LOGDS) [17].

The computational tests consider two classes of problems. The first class consists of analytical benchmark problems drawn from the literature, for which the objective and constraint functions are explicitly defined. The second class consists of constrained blackbox optimization problems, where function evaluations may be expensive and no analytical expressions are available. On both classes, ADS-PB is compared with the aforementioned methods in order to assess its performance in terms of robustness and efficiency. Comparison are done using data profiles [12, 38]. The results on analytical problems are presented in the next subsections.

4.1 Implementation in NOMAD

The ADS-PB framework is implemented within the NOMAD software as a specialized variant of the PB version of MADS-PB. This implementation was carried out in a development version of NOMAD and reuses part of the existing PB infrastructure while modifying several core components of the iteration logic.

More precisely, ADS-PB differs from the standard implementation of MADS-PB in the following aspects. First, ADS-PB does not rely on any mesh structure, and therefore no mesh projection is performed before evaluating f on trial points. Second, the set of successful points, denoted by $\mathbb{V}^k_{\text{succ}}$, is explicitly maintained throughout the optimization process. This set is used to define the punctured space $\mathring{\mathbb{R}}^n_k$. Third, unlike in the standard MADS-PB implementation, the poll centers in ADS-PB are computed only after the search step has been completed. This design is consistent with the logic of the algorithm, where the outcome of the search step may affect the incumbent solutions and therefore the points around which polling should be performed. Finally, poll points that do not belong to $\mathring{\mathbb{R}}^n_k$ are excluded from evaluation. This ensures that the poll step only generates points satisfying the exclusion criterion, in accordance with the theoretical framework introduced in this work.

4.2 Results on analytical problems from literature

The set of test problems considered in this section is taken from [5] as summarized in Table 1. It consists of fourteen constrained problems. The number of variables n ranges from 2 to 13, while the number of inequality constraints m varies from 1 to 15. For each problem, several initial points are considered, including both feasible and infeasible ones. A problem instance is defined by a given initial point together with a random seed. Each instance is solved multiple times using 10 different seeds in order to account for the stochastic components of the algorithms. This results in a total of 780 problem instances solved for each algorithm. All algorithms tested in this section are implemented within the NOMAD software. They share common search components, including a quadratic model-based search, a speculative search, and a Nelder-Mead search.

Table 1: A set of analytical constrained blackbox optimization problems [5, Table 1].

Problem	n	m	$\#x_0$	Problem	n	m	$\#x_0$
CHENWANG_F2	8	6	11	MEZMONTES	2	2	1
CHENWANG_F3	10	8	10	OPTENG_RBF	3	4	1
CRESCENT	10	2	1	PENTAGON	6	15	1
DISK	10	1	1	SNAKE	2	2	1
G210	10	2	1	SPRING	3	4	10
MAD6	5	7	10	TAOWANG_F2	7	4	10
MDO	10	10	10	ZHAOWANG_F5	13	9	10

Two variants of ADS-PB are considered, differing by the choice of poll type. These variants are compared with the corresponding versions of MADS-PB using the same poll configurations. The $2n$ variant generates a positive spanning set composed of $2n$ orthogonal directions in $\mathbb{R}^n$, as described in [1]. In contrast, the $n+1$ [13] variant constructs a set of n orthogonal directions and completes it with an additional direction obtained from a quadratic model, yielding a positive spanning set of size $n+1$. In particular, the MADS-PB $n+1$ variant corresponds to the current default implementation in `NOMAD`. For ADS-PB, the poll directions are used directly to generate trial points, without any projection onto a mesh.

Figure 3 presents the data profiles obtained on the set of analytical problems from the literature, for three target accuracies $\tau \in \{10^{-3}, 10^{-5}, 10^{-7}\}$. As the tolerance τ decreases, the convergence criterion becomes more strict, requiring higher solution accuracy. Overall, the ADS-PB variants consistently outperform their MADS-PB counterparts across all tolerances. For each poll strategy, the ADS-PB curves dominate those of MADS-PB. A similar trend is observed when comparing poll strategies. The $n+1$ variants systematically outperform the $2n$ variants for both ADS-PB and MADS-PB. This improvement can be attributed to the use of a quadratic model to generate the additional polling direction, which enhances the quality of the directions. Moreover, the $n+1$ strategy requires at most $n+1$ polling directions, compared to $2n$ for the $2n$ variant, leading to a reduced number of function evaluations per iteration. This more economical use of evaluations further contributes to its better performance. The effect is particularly pronounced at the restrictive tolerance $\tau = 10^{-7}$, where the gap between the two ADS-PB variants becomes more noticeable than for MADS-PB. One possible explanation is that ADS-PB benefits more from the model-based direction, since this direction is evaluated directly rather than projected onto a mesh, allowing the algorithm to make better use of high-quality trial points. Finally, it is worth emphasizing that ADS-PB outperforms the MADS-PB $n+1$ variant, which corresponds to the current default implementation in `NOMAD`.

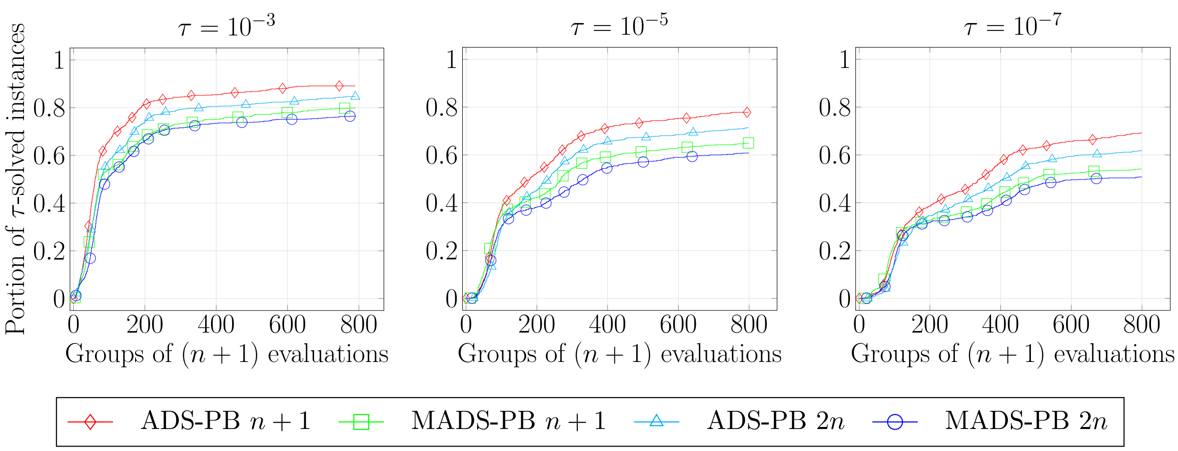


Figure 3: Data profiles on the problems of Table 1.

4.3 Results on blackbox problems

The results from the previous section were generated on a large number of analytical test problem instances. The present section focuses on time-consuming blackbox optimization problems. These problems are representative of realistic blackbox settings, where function evaluations are costly and analytical expressions are unavailable. They allow assessing the behavior of the algorithms in more challenging and practical scenarios. The experiments are conducted using ADS-PB and MADS-PB within `NOMAD`, with default configurations combining a quadratic model-based search, a speculative search, a Nelder-Mead search, and an $n+1$ poll. These methods are compared with external algorithms from the literature: NSGA-II, implemented in Python through the `pymoo` library; CMA-ES, implemented using the Python package `cma`; and LOGDS, using the MATLAB implementation made available by the authors of [17].

The first problem, referred to as STYRENE, is introduced in [4]. It corresponds to the optimization of a simulated styrene production process which involves several stages, including reactants preparation, catalytic reactions, and separation steps through distillation, with the recycling of unreacted components. The simulation follows a sequential modular approach, where each unit operation is evaluated in sequence, leading to complex dependencies between variables. The resulting optimization problem is constrained, with $n = 8$ variables and $m = 11$ inequality constraints, including four non relaxable constraints. To account for variability and robustness, this problem is solved from 30 different feasible initial points[1] and using 3 different random seeds.

The second problem is derived from the solar thermal power plant simulator introduced in [3], which is designed as a benchmark for blackbox optimization. The instance considered here is SOLAR 6, corresponding to a constrained cost minimization problem with $n = 5$ variables and $m = 6$ inequality constraints. Due to the high computational cost of function evaluations, only 30 problem instances are considered for each algorithm. These correspond to 30 initial points generated via Latin hypercube sampling[2], as provided in the associated benchmark, and a single random seed is used. The overall computational effort for the SOLAR 6 runs is approximately four CPU-days.

Figure 4 displays the data profiles obtained on the STYRENE and SOLAR 6 problems, respectively, for tolerances $\tau \in \{10^{-2}, 10^{-3}, 10^{-4}\}$. These tolerances are less strict than those considered in the previous section, reflecting the higher difficulty and computational cost of SOLAR 6.

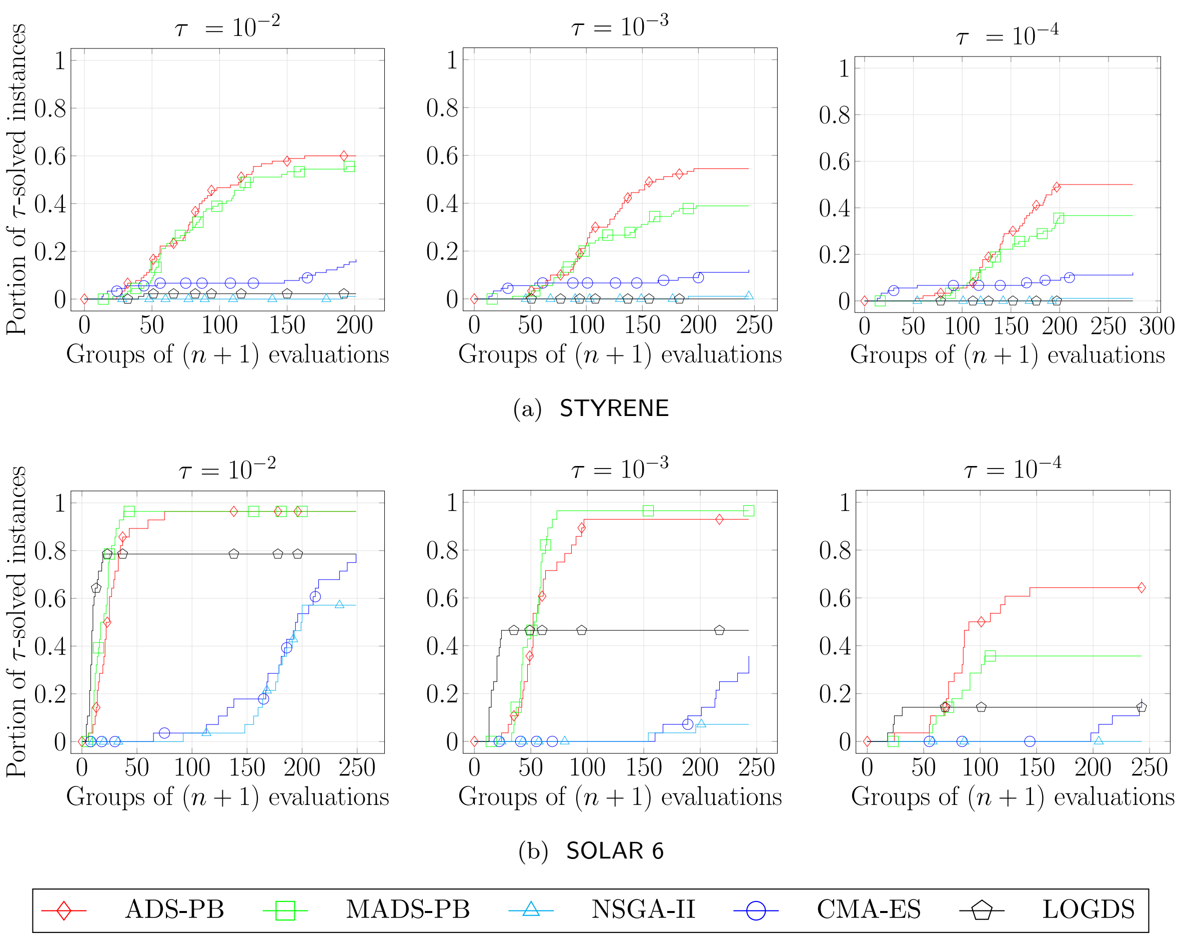


Figure 4: Data profiles on the STYRENE and SOLAR 6 problems.

[1]Starting points from https://github.com/bbopt/styrene
[2]Starting points from https://github.com/bbopt/solar

On the STYRENE problem, ADS-PB consistently outperforms MADS-PB across all tolerances, τ-solving more problems within a given computational budget. The relatively poor performance of LOGDS on this problem is explained by the fact that the four non quantifiable constraints are often violated, and blackbox evaluations often fail due to hidden constraints. Since LOGDS is not designed to handle non-relaxable and non-quantifiable constraints, infeasible evaluations provide little to no useful information, which significantly degrades its performance. The performance of CMA-ES and NSGA-II is also markedly lower, highlighting the difficulty of handling such constrained blackbox settings with general-purpose evolutionary methods. In contrast, MADS-PB remains competitive for moderate tolerances ($\tau = 10^{-2}$ and 10^{-3}), but a clear gap emerges at stricter tolerances ($\tau = 10^{-4}$), where ADS-PB shows a strong advantage.

On SOLAR 6, ADS-PB and MADS-PB again dominate the other methods, although the performance gap is less pronounced. One explanation is that LOGDS does not benefit from the full range of search and poll mechanisms available in NOMAD, such as the efficient $n+1$ poll using quadratic models to form the directions. Additionally, LOGDS appears to converge prematurely, which may limit its ability to fully exploit the available computational budget and refine solution quality. As with STYRENE, CMA-ES and NSGA-II perform significantly worse, confirming the challenge posed by these constrained blackbox problems. Overall, while MADS-PB remains competitive at moderate tolerances, ADS-PB maintains a consistent advantage when higher accuracy is required.

This improved performance of ADS-PB over MADS-PB may be attributed to the effective use of quadratic models that exploit locally smooth behaviour even when the global problem is nonsmooth, especially since search points are evaluated directly without projecting on the mesh.

5 Discussion

This work introduces ADS-PB, an extension of ADS to constrained blackbox optimization problems with quantifiable and relaxable constraints. By combining the PB mechanism with ADS, the proposed framework balances feasibility and objective function decrease while avoiding mesh projections and sufficient decrease conditions. The implementation within NOMAD enables a direct comparison with MADS-PB under identical algorithmic components.

The computational results indicate that ADS-PB can achieve improved performance, especially at higher accuracy levels, likely due to its ability to directly evaluate high-quality trial points. Beyond these results, ADS-PB provides a flexible structure that can accommodate model-based and hybrid strategies while maintaining convergence guaranties under standard assumptions.

Future work will investigate hybrid strategies that combine ADS with sufficient decrease mechanisms to possibly derive complexity guarantees. Moreover, another direction is to develop a fusion of MADS-PIP [5] and ADS-PB, with the aim of efficiently leveraging both relaxable equality and inequality constraints. Another promising direction is the design of new search strategies that explicitly exploit the absence of mesh projection in ADS-PB, together with an analysis of their theoretical impact.

References

[1] M.A. Abramson, C. Audet, J.E. Dennis, Jr., and S. Le Digabel. OrthoMADS: A Deterministic MADS Instance with Orthogonal Directions. SIAM Journal on Optimization, 20(2):948–966, 2009. doi:10.1137/080716980.

[2] S. Alarie, C. Audet, A.E. Gheribi, M. Kokkolaras, and S. Le Digabel. Two decades of blackbox optimization applications. EURO Journal on Computational Optimization, 9:100011, 2021. doi:10.1016/j.ejco.2021.100011.

[3] N. Andrés-Thió, C. Audet, M. Diago, A.E. Gheribi, S. Le Digabel, X. Lebeuf, M. Lemyre Garneau, and C. Tribes. Solar: a solar thermal power plant simulator for blackbox optimization benchmarking. Optimization and Engineering, 26(3):1815–1861, 2025. doi:10.1007/s11081-024-09952-x.

[4] C. Audet, V. Béchard, and S. Le Digabel. Nonsmooth optimization through Mesh Adaptive Direct Search and Variable Neighborhood Search. Journal of Global Optimization, 41(2):299–318, 2008. doi:10.1007/s10898-007-9234-1.

[5] C. Audet, A. Brilli, Y. Diouane, S. Le Digabel, E.J. Silva, and C. Tribes. A penalty-interior point method combined with MADS for equality and inequality constrained optimization. Technical Report 2601.20811, arXiv, 2026. URL: https://doi.org/10.48550/arXiv.2601.20811.

[6] C. Audet, A.R. Conn, S. Le Digabel, and M. Peyrega. A progressive barrier derivative-free trust-region algorithm for constrained optimization. Computational Optimization and Applications, 71(2):307–329, 2018. doi:10.1007/s10589-018-0020-4.

[7] C. Audet and J.E. Dennis, Jr. Analysis of Generalized Pattern Searches. SIAM Journal on Optimization, 13(3):889–903, 2003. doi:10.1137/S1052623400378742.

[8] C. Audet and J.E. Dennis, Jr. Mesh Adaptive Direct Search Algorithms for Constrained Optimization. SIAM Journal on Optimization, 17(1):188–217, 2006. doi:10.1137/040603371.

[9] C. Audet and J.E. Dennis, Jr. A Progressive Barrier for Derivative-Free Nonlinear Programming. SIAM Journal on Optimization, 20(1):445–472, 2009. doi:10.1137/070692662.

[10] C. Audet, T. Denorme, Y. Diouane, S. Le Digabel, and C. Tribes. Adaptive direct search algorithms for constrained optimization. Technical Report G-2025-53, Les cahiers du GERAD, 2025. doi:10.48550/arXiv.2507.23054.

[11] C. Audet and W. Hare. Derivative-Free and Blackbox Optimization. Springer Series in Operations Research and Financial Engineering. Springer International Publishing, Cham, Switzerland, second edition, 2026. doi:10.1007/978-3-032-00906-7.

[12] C. Audet, W. Hare, and C. Tribes. A summary of benchmarking constrained, multi-objective and surrogate-assisted optimization methods. Optimization Letters, 2026. doi:10.1007/s11590-026-02302-z.

[13] C. Audet, A. Ianni, S. Le Digabel, and C. Tribes. Reducing the Number of Function Evaluations in Mesh Adaptive Direct Search Algorithms. SIAM Journal on Optimization, 24(2):621–642, 2014. doi:10.1137/120895056.

[14] C. Audet, S. Le Digabel, V. Rochon Montplaisir, and C. Tribes. Algorithm 1027: NOMAD version 4: Nonlinear optimization with the MADS algorithm. ACM Transactions on Mathematical Software, 48(3):35:1–35:22, 2022. doi:10.1145/3544489.

[15] C. Audet and C. Tribes. Mesh-based Nelder-Mead algorithm for inequality constrained optimization. Computational Optimization and Applications, 71(2):331–352, 2018. doi:10.1007/s10589-018-0016-0.

[16] A.S. Berahas, O. Sohab, and L.N. Vicente. Full-low evaluation methods for derivative-free optimization. Optimization Methods and Software, 38(2):386–411, 2023. doi:10.1080/10556788.2022.2142582.

[17] A. Brilli, A.L. Custódio, G. Liuzzi, and E.J. Silva. Nonlinear derivative-free constrained optimization with a penalty-interior point method and direct search. Numerical Algorithms, 2026. doi:10.1007/s11075-026-02360-5.

[18] C. Cartis and L. Roberts. Randomized subspace derivative-free optimization with quadratic models and second-order convergence. Optimization Methods and Software, pages 1–28, 2026. doi:10.1080/10556788.2025.2601668.

[19] F.H. Clarke. Optimization and Nonsmooth Analysis. John Wiley and Sons, New York, 1983. Reissued in 1990 by SIAM Publications, Philadelphia, as Vol. 5 in the series Classics in Applied Mathematics. URL: http://www.ec-securehost.com/SIAM/CL05.html.

[20] A.R. Conn and S. Le Digabel. Use of quadratic models with mesh-adaptive direct search for constrained black box optimization. Optimization Methods and Software, 28(1):139–158, 2013. doi:10.1080/10556788.2011.623162.

[21] A.R. Conn, K. Scheinberg, and L.N. Vicente. Introduction to Derivative-Free Optimization. MOS-SIAM Series on Optimization. SIAM, Philadelphia, 2009. doi:10.1137/1.9780898718768.

[22] A.L. Custódio, K. Scheinberg, and L.N. Vicente. Methodologies and software for derivative-free optimization. In T. Terlaky, M.F. Anjos, and S. Ahmed, editors, Advances and Trends in Optimization with Engineering Applications, MOS-SIAM Book Series on Optimization, chapter 37. SIAM, Philadelphia, 2017. URL: http://www.mat.uc.pt/~lnv/papers/dfo-survey.pdf.

[23] C. Davis. Theory of positive linear dependence. American Journal of Mathematics, 76:733–746, 1954. URL: http://www.ams.org/mathscinet-getitem?mr=16:211e.

[24] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J.J. Merelo, and Hans-Paul H.P. Schwefel, editors, Parallel Problem Solving from Nature PPSN VI, pages 849–858, Berlin, Heidelberg, 2000. Springer. doi:10.1007/3-540-45356-3_83.

[25] Y. Diouane. A merit function approach for evolution strategies. EURO Journal on Computational Optimization, 9:100001, 2021. doi:10.1016/j.ejco.2020.100001.

[26] Y. Diouane, V. Picheny, R. Le Riche, and A. Scotto Di Perrotolo. TREGO: a trust-region framework for efficient global optimization. Journal of Global Optimization, pages 1–23, 2022. doi:10.1007/s10898-022-01245-w.

[27] K.J. Dzahini, F. Rinaldi, C.W. Royer, and D. Zeffiro. Direct-search methods in the year 2025: Theoretical guarantees and algorithmic paradigms. EURO Journal on Computational Optimization, 13:100110, 2025. doi:10.1016/j.ejco.2025.100110.

[28] S. Gratton and L.N. Vicente. A Merit Function Approach for Direct Search. SIAM Journal on Optimization, 24(4):1980–1998, 2014. doi:10.1137/130917661.

[29] R. Gupta and Q. Zhang. Data-driven decision-focused surrogate modeling. AIChE Journal, 70(4):e18338, 2024. doi:10.1002/aic.18338.

[30] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. Evolutionary computation, 9(2):159–195, 2001. doi:10.1162/106365601750190398.

[31] J. Jahn. Introduction to the Theory of Nonlinear Optimization. Springer Cham, 4th edition, 2020. doi:10.1007/978-3-030-42760-3.

[32] E. Karantoumanis and N. Ploskas. Improving derivative-free optimization algorithms through an adaptive sampling procedure. Results in Control and Optimization, 16:100460, 2024. doi:10.1016/j.rico.2024.100460.

[33] T.G. Kolda, R.M. Lewis, and V. Torczon. Optimization by direct search: New perspectives on some classical and modern methods. SIAM Review, 45(3):385–482, 2003. doi:10.1137/S003614450242889.

[34] J. Larson, M. Menickelly, and S.M. Wild. Derivative-free optimization methods. Acta Numerica, 28:287–404, 2019. doi:10.1017/S0962492919000060.

[35] S. Le Digabel and S.M. Wild. A taxonomy of constraints in black-box simulation-based optimization. Optimization and Engineering, 25(2):1125–1143, 2024. doi:10.1007/s11081-023-09839-3.

[36] Y. Liu and Y. Li. A Model-Based Derivative-Free Optimization Algorithm for Partially Separable Problems. Technical Report 2506.21948, arXiv, 2025. URL: https://doi.org/10.48550/arXiv.2506.21948.

[37] K. Ma, L.M. Rios, H. Zheng, N.V. Sahinidis, and S. Rajagopalan. Model-and-search: a derivative-free local optimization algorithm. Computational Optimization and Applications, 92(3):889–921, 2025. doi:10.1007/s10589-025-00686-9.

[38] J.J. Moré and S.M. Wild. Benchmarking Derivative-Free Optimization Algorithms. SIAM Journal on Optimization, 20(1):172–191, 2009. doi:10.1137/080724083.

[39] R.T. Rockafellar. Generalized directional derivatives and subgradients of nonconvex functions. Canadian Journal of Mathematics, 32(2):257–280, 1980.