GLODS: GLOBAL AND LOCAL OPTIMIZATION USING DIRECT SEARCH

A. L. CUSTÓDIO * AND J. F. A. MADEIRA †

Abstract. Locating and identifying points as global minimizers is, in general, a hard and time-consuming task. Difficulties increase in the impossibility of using the derivatives of the functions defining the problem. In this work, we propose a new class of methods suited for global derivative-free constrained optimization. Using direct search of directional type, the algorithm alternates between a search step, where potentially good regions are located, and a poll step where the previously located promising regions are explored. This exploitation is made through the launching of several instances of directional direct searches, one in each of the regions of interest. Differently from a simple multistart strategy, direct searches will merge when sufficiently close. The goal is to end with as many direct searches as the number of local minimizers, which would easily allow locating the global extreme value. We describe the algorithmic structure considered, present the corresponding convergence analysis and report numerical results, showing that the proposed method is competitive with currently commonly used global derivative-free optimization solvers.

Key words. Global optimization, multistart strategies, direct-search methods, pattern-search methods, nonsmooth calculus.

AMS subject classifications. 90C56, 90C26, 90C30.

1. Introduction.

Let us consider the global minimization problem defined as:

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & x \in \Omega \subset \mathbb{R}^n, \end{aligned}$$

where $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ represents a real-extended value function and $\Omega \subset \mathbb{R}^n$ a compact set, defining the problem feasible region. GLODS (Global and Local Optimization using Direct Search) class is designed for computing all the problem local minima, from which the global minimum would be easily identified. For the application of the method, no assumptions regarding the smoothness of the functions defining the problem are required. Nevertheless, in order to guarantee the existence of a global minimum, f should at least be lower-semicontinuous in Ω.

Solving global optimization problems is a challenging task, with additional difficulties when derivatives are not available for use. Although, there is a large number of practical real-world applications where global derivative-free optimization is required, in domains that vary from electrical engineering (for example, in the design of hybrid electric vehicles [13] or for reinforcement learning in robotics [25]) to chemistry (molecular conformal optimization problems [2]), acoustics [29] or multidisciplinary design optimization [33].

Typical approaches of derivative-free optimization to global problems consider a partition of the feasible region into subdomains, which are locally explored by a derivative-free algorithm. This is the case of DIRECT [19] and MCS [16], in the latter

*Department of Mathematics, FCT-UNL-CMA, Quinta da Torre, 2829-516 Caparica, Portugal (alcustodio@fct.unl.pt). Support for this author was provided by Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) under the project PEst-OE/MAT/UI0297/2014 (CMA) and the grant PTDC/MAT/116736/2010.

†IDMEC, Instituto Superior Técnico, Universidade de Lisboa, Av. Rovisco Pais, 1040-001 Lisbon, Portugal and Department of Mathematics, ISEL, Rua Conselheiro Emídio Navarro, 1, 1959-007 Lisbon, Portugal (aguilarmadeira@tecnico.ulisboa.pt). Support for this author was provided by ISEL, IPL, Lisboa, Portugal and LAETA, IDMEC, Instituto Superior Técnico, Universidade de Lisboa, Lisboa, Portugal.

with the use of quadratic polynomial interpolation to enhance the local optimization. Convergence to global minimizers results from the density of the sample sets of points considered.

Other strategies propose hybrid versions of common heuristics and derivative--free local algorithms, like is the case of simulated annealing, tabu search, evolution strategies, particle swarm optimization or variable neighborhood search (see [14, 15, 32, 38, 3], respectively). Heuristics attempt to confer to the algorithms some global behavior. Convergence results, when available [32, 38, 3], are a consequence of the properties of the local optimization algorithms.

When practitioners are faced with the need of computing a global minimum, a common approach consists in using a local optimization procedure coupled with a multistart strategy. Pure multistart strategies are generally quite inefficient, since several local searches will converge to the same minimizer. In a derivative-free optimization setting, enhancements to multistart can include the use of response surface models [34] or, in a directional direct search context, the assessment of variability to dynamically adapt the number of poll directions [24].

GLODS, when the search step is defined by a deterministic procedure (like the 2^n-Centers strategy described in Section 2.1 or when using Sobol or Halton sequences), intends to be a clever deterministic alternative to pure multistart. The algorithm will consider a local search procedure based on direct search of directional type [10], where each generated point will have associated a comparison radius. This comparison radius is crucial for improving the efficiency of multistart, since it will allow the merging of direct searches considered to be sufficiently close. Differently from Multilevel Single Linkage [20, 21, 26], one well known stochastic algorithm also based in multistart, the proposed comparison radius is related to the step size parameter used by the algorithm, not depending on any probabilistic considerations.

For simplicity, and similarly to other derivative-free optimization algorithms, an extreme barrier approach will be adopted, replacing f by the extreme barrier function f_Ω defined as:

$$f_\Omega(x) = \begin{cases} f(x) & \text{if } x \in \Omega, \\ +\infty & \text{otherwise.} \end{cases} \tag{1.1}$$

Infeasible points will not be evaluated, being the corresponding objective function value set equal to $+\infty$. Other possibilities to deal with constraints could include filter methods [5] or progressive barrier approaches [7].

Section 2 will describe the motivation behind the algorithmic design, also providing a rigorous description of GLODS. Using Clarke [9] nonsmooth analysis, Section 3 will establish the convergence properties of the method. Numerical experiments in a set of bound constrained global optimization problems are reported in Section 4, being numerical results detailed in Appendix A. The paper ends in Section 5 with some remarks.

2. GLODS: Global and Local Optimization using Direct Search. Like in a classical direct-search method of directional type, the algorithmic structure of GLODS is organized around a search and a poll step. The main goal of the search step is to explore the whole feasible region, in an attempt to locate good promising subdomains, which would then be locally explored by the poll step of the algorithm. The poll step is responsible for ensuring the convergence of the method, but the

quality of the computed minimizers, as corresponding to local or global minima, will depend on the search step.

A list L_k, storing feasible points in tuples of form $(x; \alpha; r; i)$, will be kept during the optimization process. Points generated either at the search or poll steps, corresponding to the sets A_k and P_k, respectively, could be added to this list. Each new point, x, is stored jointly with the corresponding step size parameter, α, the comparison radius, r, and its classification as active or inactive, corresponding to setting the index i equal to 1 or 0, respectively. In a crude definition, a point is classified as active when it presents the best objective function value among all the points sufficiently close to it. Closeness will be measured by the comparison radius, r. During the course of the optimization, an active point could change its status to inactive (never the opposite). Only active points will be selected as poll centers.

Several methods could be considered for defining the search step in GLODS, and consequently the trial list A_k: random sampling [35], Latin hypercube sampling [27], Sobol sequences or Halton sequences [22]. All these strategies have in common the fact of generating asymptotically dense sets of points in a compact set. We considered an additional strategy entitled 2^n-Centers, which will be detailed in Section 2.1.

The poll step starts by ordering the active points in the list, and selecting one as the new poll center. A local exploitation of the region around this poll center will be conducted by testing the directions belonging to a positive spanning set or a positive basis [11], scaled by the step size parameter, α. In this procedure, complete or opportunistic approaches could be taken.

A general schematic description of the method can be found in Algorithm 2.1.

ALGORITHM 2.1 (**GLODS: Global and Local Optimization using Direct Search**).

Initialization

Let $\mathcal{D}$ be a (possibly infinite) set of positive spanning sets, such that $\forall d \in D \in \mathcal{D}, 0 < d_{min} \leq \|d\| \leq d_{max}$. Choose $r_0 \geq d_{max}\alpha_0 > 0$, $0 < \beta_1 \leq \beta_2 < 1$, and $\gamma \geq 1$. Set $L_0 = \emptyset$.

For $k = 0, 1, 2, \ldots$

1. **Search step:** Compute a finite set of distinct points $A_k = \{(x_j; 0; 0; 0) : f_\Omega(x_j) < +\infty\}$ (all $x_j \in A_k$ should be in a mesh if $\bar{\rho}(\cdot) \equiv 0$, see Section 3.1). Call L_k `= add(`L_k`,`A_k`)` to possibly add some new points in A_k to L_k. If $k = 0$, go to the poll step. Otherwise, if there is a new active point in L_k then set $L_{k+1} = L_k$, declare the iteration (and the search step) as successful and skip the poll step.

2. **Poll step:** Order the list L_k and select an active point $(x; \alpha_x; r_x; 1) \in L_k$ as the current iterate, corresponding step size parameter, and comparison radius (thus setting $(x_k; \alpha_k; r_k; i_k) = (x; \alpha_x; r_x; 1)$).
Choose a positive spanning set D_k from the set $\mathcal{D}$. Compute the set of poll points $P_k = \{(x_k + \alpha_k d; \alpha_k; \alpha_k\|d_k\|; 0) : d \in D_k \wedge f_\Omega(x_k + \alpha_k d) < +\infty\}$. Call L_{k+1} `= add(`L_k`,`P_k`)` to possibly add some new points in P_k to L_k. If there is a new active point in L_{k+1} declare the iteration (and the poll step) as successful. If no new point was added to L_k declare the iteration (and the poll step) as unsuccessful. Otherwise declare the iteration (and the poll step) as merging.

3. **Step size parameter and radius update:** If the iteration was successful then maintain or increase the corresponding step size parameters: $\alpha_{new} \in [\alpha, \gamma\alpha]$ and replace all the new active points $(x; \alpha_x; r_x; 1)$ in L_{k+1} by $(x; \alpha_{new}; d_{max}\alpha_{new}; 1)$, if $d_{max}\alpha_{new} > r_x$, or by $(x; \alpha_{new}; r_x; 1)$, when $d_{max}\alpha_{new} \leq r_x$.
   If the iteration was unsuccessful then decrease the corresponding step size parameter: $\alpha_{new} \in [\beta_1\alpha_k, \beta_2\alpha_k]$ and replace the poll point $(x_k; \alpha_k; r_k; 1)$ in L_{k+1} by $(x_k; \alpha_{new}; r_k; 1)$.

The description provided in Algorithm 2.1 is deliberately close to the one of a classical direct-search method of directional type. However, since the main goal of the search step is to identify promising subdomains, there is no need to perform it at every iteration. Several strategies could be implemented regarding this decision, possibly incorporating some user knowledge about the problem under analysis. For instance, if the user has an idea about the minimum number of local minimizers, the search step could be executed at each iteration where the number of active points in the list is less or equal than this value.

Algorithm 2.2 manages the list of feasible points. A point is added to the list as an active point under one of the two following conditions:

- its distance to any of the points already stored exceeds the corresponding comparison radius, meaning that the new point belongs to a part of the feasible region not yet explored (line numbered as 1 in Algorithm 2.2);
- the new point, x_{new}, is comparable with at least one active point, y, meaning $\|x_{new} - y\| \leq r_y$ (line numbered as 3 in Algorithm 2.2), decreases the corresponding objective function value, meaning $f(x_{new}) < f(y) - \bar{\rho}(\alpha_y)$ (line numbered as 4 in Algorithm 2.2, when i_{dom} is set equal to 1), and no point to which it is comparable with equals or decreases the corresponding objective function value, $f(y) \not\leq f(x_{new}) - \bar{\rho}(\alpha_y)$ (line numbered as 5 in Algorithm 2.2).

The function $\bar{\rho}(.)$ represents the constant zero function, if a simple decrease approach is taken, or a forcing function $\rho : (0, +\infty) \to (0, +\infty)$, i.e., a continuous and non-decreasing function, satisfying $\rho(t)/t \to 0$ when $t \downarrow 0$ (see [23]), when a sufficient decrease strategy is adopted. Typical examples of forcing functions are $\rho(t) = t^{1+a}$, for $a > 0$.

Additionally, in a simple decrease approach, a point could be added to the list as active if it decreases the objective function value of any point (not necessarily active) to which it is comparable with (line numbered as 6 in Algorithm 2.2, when $p_{dom} = 0$ and $i_{comp} = 1$).

Inactive points can also be added to the list. This situation occurs when a new point is comparable with an already stored active point, presenting a better objective function value than it, meaning

$$f(x_{new}) < f(y) - \bar{\rho}(\alpha_y), \tag{2.1}$$

(line numbered as 4 in Algorithm 2.2, when i_{dom} is set equal to 1)

but another point already stored, comparable with the new one, equals or decreases this value (line numbered as 5 in Algorithm 2.2).

When adding a new point to the list, the algorithm defines the corresponding step size parameter and comparison radius. If the point is in a new subdomain of the feasible region, meaning that its distance to any of the points already stored exceeds

the corresponding comparison radius, these values will be the ones considered for initialization (line numbered as 2 in Algorithm 2.2). Otherwise, if generated at the poll step, the step size parameter will be equal to the one of the poll center and the comparison radius to the step size of the poll center times the norm of the corresponding poll direction (line numbered as 8 in Algorithm 2.2). When generated in the search step, the new point inherits the parameters of the point presenting the largest step size, comparable with it, for which equation (2.1) holds (line numbered as 7 in Algorithm 2.2). A schematic description of these procedures is detailed in Algorithm 2.2.

Algorithm 2.2: $[L_1]$=`add`(L_1, L_2)
Procedure for adding new points, stored in L_2, to the current list, L_1.

forall the $(x; \alpha_x; r_x; 0) \in L_2$ **do**
1 **if** $\min_{y \in L_1} (\|x - y\| - r_y) > 0$ **then**
2 $L_1 = L_1 \cup \{(x; \alpha_0; r_0; 1)\}$
else
if $x \notin L_1$ **then**
set $\alpha_a = 0$, $r_a = 0$, $i_{dom} = 0$, $p_{dom} = 0$ and $i_{comp} = 0$
forall the $(y; \alpha_y; r_y; i_y) \in L_1$ **do**
3 **if** $\|x - y\| - r_y \leq 0$ **then**
4 **if** $f(x) < f(y) - \bar{\rho}(\alpha_y)$ **then**
$i_{comp} = 1$
$i_{dom} = i_{dom} + i_y$
$i_y = 0$
if $\alpha_y > \alpha_a$ **then**
$\alpha_a = \alpha_y$
$r_a = r_y$
end
else
5 **if** $f(y) \leq f(x) - \bar{\rho}(\alpha_y)$ **then**
$p_{dom} = 1$
end
end
end
end
if $p_{dom} = 0$ **then**
$i_x = 1$
end
6 **if** $i_{dom} > 0 \vee (\bar{\rho}(.) \equiv 0 \wedge p_{dom} = 0 \wedge i_{comp} = 1)$ **then**
if $\alpha_x = 0$ **then**
7 $L_1 = L_1 \cup \{(x; \alpha_a; r_a; i_x)\}$
else
8 $L_1 = L_1 \cup \{(x; \alpha_x; r_x; i_x)\}$
end
end
end
end
end

The procedure described in Algorithm 2.2 is responsible for the distinction between GLODS and the use of a classical direct-search method of directional type coupled with a multistart strategy. In GLODS, if two distinct points are sufficiently close, meaning to a distance equal or inferior to the corresponding comparison radius, only one of the two could remain active in the list (when $\bar{\rho}(.) \equiv 0$). This procedure allows the merging of direct searches with different initializations, when sufficiently close to each other.

Also different from a classic direct-search method of directional type, GLODS generates three types of iterations: *successful*, *unsuccessful*, and *merging*. A *successful iteration* corresponds to at least one new active point added to the list. When no points are added to the list, the iterate is named as *unsuccessful*. *Merging iterations* occur when only adding inactive points.

At unsuccessful iterations, the step size parameter corresponding to the poll center is obligatory decreased. At successful iterations the step sizes corresponding to the new active points found could be increased (or kept constant). Merging iterations do not imply changes in step sizes. In order to allow the comparison between the poll points and the poll center, the comparison radius should at least be equal to the step size parameter times the maximum norm of the poll directions. Thus, if after a successful iteration the update of the step size parameter prevents the comparison between the poll center and the poll points, the comparison radius will be increased to an adequate value.

2.1. Search step based in 2^n-Centers. In the previous section it was already mentioned that the main purpose of the search step is to identify promising subdomains of the feasible region, to be locally explored by the poll step of the algorithm. Thus, when defining a strategy for the search step, ideally it should:

- generate a set of points asymptotically dense in the feasible region: ensuring that asymptotically all the feasible region will be explored;
- cover the feasible region in a similar way: meaning that without evidence of function decrease, no primacy should be given to some parts of the feasible region;
- avoid randomness: guaranteeing equal algorithmic performance for different runs, in the same problem.

The 2^n-Centers strategy complies with these three requirements, being a possibility for defining a deterministic search step in GLODS. It starts by enclosing the feasible region in a box, which is going to be consecutively subdivided into smaller boxes, defining different levels of search. At each level, the points to be selected for evaluating the objective function correspond to the box centers. Exception occurs at level 0, where all the vertices of the box are additionally considered. Figure 2.1 exemplifies the procedure for a bound constrained problem of dimension $n = 2$.

At level $\ell > 0$, the number of box centers is equal to $2^{n\times\ell}$. For problems with a large value of n, or for high levels of search, this could represent a considerable computational effort, once that each of these box centers could remain active, after application of Algorithm 2.2, thus corresponding to an initialization for a new direct search. A possible strategy to partially circumvent this problem is, at each execution of the search step, to select a subset of the box centers not yet considered, of dimension 2^n, with one box center located in each of the new boxes generated in the current level. Exception again occurs in level 0, where the $2^n + 1$ points are jointly considered. The remaining box centers will be used in the following iterations where the search step would be performed (each time by considering a subset of dimension 2^n). Figure 2.2

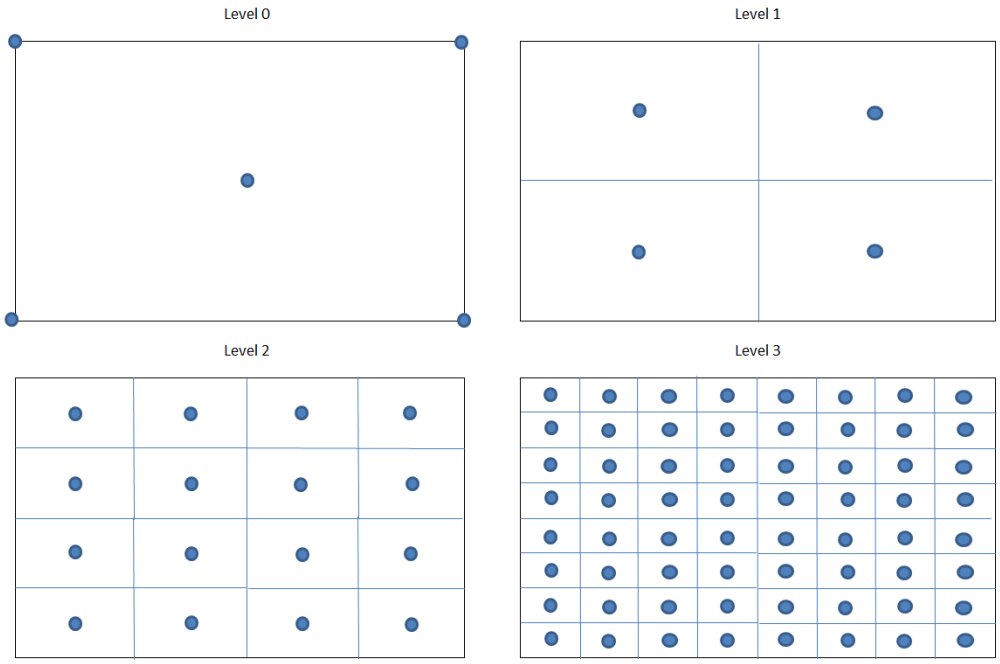

FIG. 2.1. *Four different levels of the strategy* 2^n-Centers *for a bound constrained problem of dimension* $n = 2$.

illustrates the procedure for $n = 2$ and $l = 2$.

3. Convergence analysis. In a classical direct-search method of directional type there is always the guarantee that at the end of each iteration there will be a poll center to proceed with the optimization. In GLODS, since poll centers are selected from active points, the existence of merging iterations, where no active points are added to the list and some active points change the corresponding status to inactive, justifies the need of establishing the following proposition.

PROPOSITION 3.1. *At the end of each iteration of Algorithm 2.1, at least one element* $z \in argmin_{w \in L} f(w)$ *is active.*

Proof. Suppose not. Let $z \in \operatorname{argmin}_{w \in L} f(w)$ be one element of the considered set, computed at the end of the current iteration. The point z could have been added to the list, during the current iteration, as inactive or, being an active point already in the list, changed its status to inactive.

In the latter situation, there should have been a point x such that $\|x - z\| \leq r_z$ and $f(x) < f(z) - \bar{\rho}(\alpha_z) \leq f(z)$. Since z was active, x will be added to the list, leading to a contradiction.

In the former situation, there should have been $y \in L$, satisfying $\|z - y\| \leq r_y$, $f(z) \geq f(y) - \bar{\rho}(\alpha_y)$ and $f(y) \leq f(z) - \bar{\rho}(\alpha_y)$. If $f(y) < f(z)$ then we have arrived to a contradiction. If $f(y) = f(z)$ and y is active then we have contradicted the initial assumption. When $f(y) = f(z)$ and y is inactive, a recursive argument, following the steps presented, will also allow us to arrive to a contradiction. □

The convergence analysis of classical direct-search methods of directional type starts by establishing the existence of a subsequence of step size parameters converging

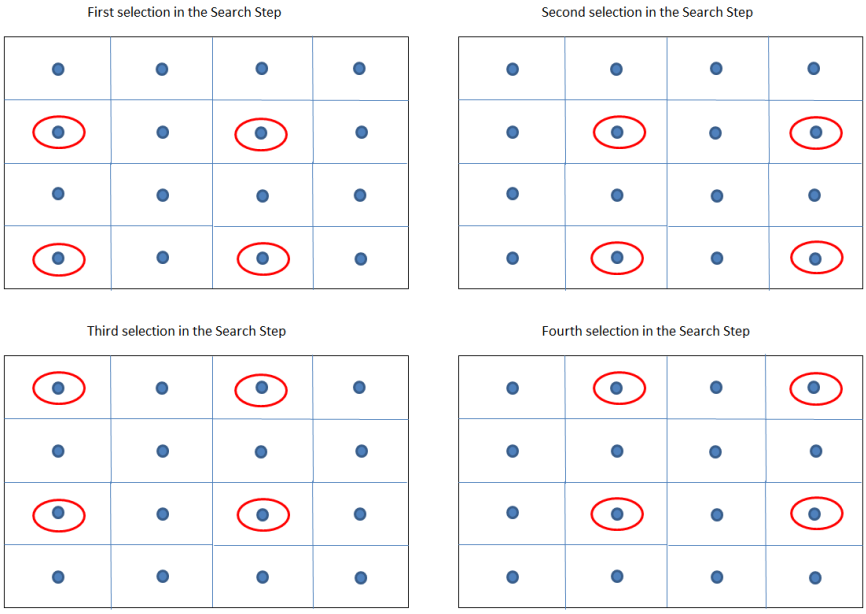

FIG. 2.2. *Points generated at several executions of a search step defined by the* 2^n-Centers *strategy, for a bound constrained problem of dimension* $n = 2$ *and a search level* $l = 2$.

to zero. Two major strategies can be considered to enforce this property: the use of integer lattices (like is the case of Generalized Pattern Search (GPS) [4] or Mesh Adaptive Direct Search (MADS) [6]) or by imposing a sufficient decrease condition, through the definition of $\bar{\rho}(.)$ as a forcing function (similarly to what is done in Generating Set Search [23]).

With this goal, let us consider the following assumptions.

ASSUMPTION 3.1. *The set* $\Omega \subset \mathbb{R}^n$ *is compact.*

ASSUMPTION 3.2. *The function* f *is lower bounded in* $\Omega \subset \mathbb{R}^n$.

3.1. Using integer lattices. The level of smoothness present in the objective function has implications in the type of positive spanning sets that could be considered in the definition of the poll step of the algorithm. For continuously differentiable functions, a finite set of directions which satisfies appropriate integrality requirements is enough [4, 23].

ASSUMPTION 3.3. *The set* $\mathcal{D} = D$ *of positive spanning sets is finite and the elements of* D *are of the form* $G\bar{z}_j$, $j = 1, \ldots, |D|$, *where* $G \in \mathbb{R}^{n\times n}$ *is a nonsingular matrix and each* $\bar{z}_j$ *is a vector in* $\mathbb{Z}^n$.

In the presence of nondifferentiabilities, it is advisable to consider an infinite set of positive spanning sets $\mathcal{D}$, which should be dense (after normalization) in the unit sphere. Additionally, some care must be taken when computing the set $\mathcal{D}$, in particular to guarantee that the points generated by the algorithm lie in an integer lattice [6].

ASSUMPTION 3.4. *Let* D *represent a finite set of positive spanning sets satisfying Assumption 3.3.*

The set $\mathcal{D}$ *is so that the elements* $d_k \in D_k \in \mathcal{D}$ *satisfy the following conditions:*

1. d_k *is a nonnegative integer combination of the columns of* D.

2. *The distance between x_k and the point $x_k + \alpha_k d_k$ tends to zero if and only if α_k does:*

$$\lim_{k \in K} \alpha_k \|d_k\| \;=\; 0 \iff \lim_{k \in K} \alpha_k \;=\; 0,$$

for any infinite subsequence K.

3. *The limits of all convergent subsequences of $\bar{D}_k = \{d_k/\|d_k\| : \; d_k \in D_k\}$ are positive spanning sets for $\mathbb{R}^n$.*

The third requirement above is part of the MADS original presentation [6], but is not used in the convergence analysis for nonsmooth objective functions.

Regarding the step size parameter updates, some form of rationality should also be ensured.

ASSUMPTION 3.5. *Let $\tau > 1$ be a rational number and $m^{max} \geq 0$ and $m^{min} \leq -1$ integers. If the iteration is successful, then the step size parameter is maintained or increased by considering $\alpha_{new} = \tau^{m^+}\alpha$, with $m^+ \in \{0, \ldots, m^{max}\}$. If the iteration is unsuccessful, then the step size parameter is decreased by setting $\alpha_{new} = \tau^{m^-}\alpha$, with $m^- \in \{m^{min}, \ldots, -1\}$.*

The updating strategy described in Algorithm 2.1 conforms to the one of Assumption 3.5 by setting $\beta_1 = \tau^{m^{min}}$, $\beta_2 = \tau^{-1}$, and $\gamma = \tau^{m^{max}}$.

All the points generated by the algorithm, either in the search step or the poll step, should lie in an implicitly defined mesh. Considering the mesh definition given in Assumption 3.6 below, this condition is trivially satisfied for points generated during the polling procedure.

ASSUMPTION 3.6. *At iteration k, the search step in Algorithm 2.1 only evaluates points in*

$$M_k \;=\; \bigcup_{x \in E_k} \{x + \alpha_k D z : z \in \mathbb{N}_0^{|D|}\},$$

where E_k represents the set of all the points evaluated by the algorithm previously to iteration k.

The previous assumptions allow us to derive the first result, required for establishing the convergence of GLODS, namely the existence of a subsequence of step size parameters converging to zero. This result was originally established by Torczon [37], in the context of pattern search, and generalized by Audet and Dennis to GPS [4] and MADS [6].

THEOREM 3.2. *Let Assumption 3.1 hold. Algorithm 2.1 under one of the Assumptions 3.3 or 3.4 combined with Assumptions 3.5–3.6 and $\bar{\rho}(\cdot) \equiv 0$ generates a sequence of iterates satisfying*

$$\liminf_{k \to +\infty} \alpha_k \;=\; 0.$$

Proof. Let us assume that there is α_* such that $\alpha_k > \alpha_* > 0, \forall k$. Assumptions 3.3 or 3.4 combined with Assumptions 3.5–3.6 ensure that all points generated by Algorithm 2.1 lie in an integer lattice (see [4, 6]). The intersection of a compact set with an integer lattice is finite. Since Ω is compact, there is only a finite number of different points that could be generated by the algorithm. Successful or merging iterations correspond to at least one new feasible point added to the list. Once a point is added to this list it will always remain on it (possibly changing its status to inactive). Thus, the number of successful and merging iterations must be finite, and

consequently there is an infinite number of unsuccessful iterations. The step size at unsuccessful iterations is reduced by at least $\beta_2 \in]0,1[$, which contradicts the existence of a lower bound for the step size parameter. □

3.2. Imposing sufficient decrease. A different strategy, to achieve a similar result to the one stated in Theorem 3.2, consists in defining a forcing function, $\bar{\rho}(\cdot) = \rho(\cdot)$, requiring sufficient rather than simple decrease for accepting a new point. Let us consider the additional assumption, part of Assumption 3.4.

ASSUMPTION 3.7. *The distance between x_k and the point $x_k + \alpha_k d_k$ tends to zero if and only if α_k does:*

$$\lim_{k \in K} \alpha_k \|d_k\| \;=\; 0 \iff \lim_{k \in K} \alpha_k \;=\; 0,$$

for all $d_k \in D_k$ and for any infinite subsequence K.

The following result was first derived by Kolda, Lewis and Torczon [23] in the context of Generating Set Search.

THEOREM 3.3. *Let Assumptions 3.1–3.2 hold. Algorithm 2.1, when $\bar{\rho}(\cdot)$ is a forcing function and Assumption 3.7 holds, generates a sequence of iterates satisfying*

$$\liminf_{k \to +\infty} \alpha_k \;=\; 0.$$

Proof. Assume that there is α_* such that $\alpha_k > \alpha_* > 0, \forall k$. Let us start by showing that there is an infinite number of successful iterations.

Suppose not. Active points are added to the list only at successful iterations. Thus, the number of active points in the list must be finite. Suppose there is an infinite number of merging iterations. At each merging iteration at least one of the active points in the list changes its status to inactive. This contradicts the fact of having a finite number of active points in the list.

Let us now assume that there is a finite number of successful and merging iterations, being infinite the number of unsuccessful iterations. At each unsuccessful iteration the step size parameter of the corresponding active poll center is reduced by at least $\beta_2 \in]0,1[$, which contradicts the existence of the lower bound $\alpha_* > 0$ for the step size parameter.

The previous arguments allow us to conclude that there is an infinite number of successful iterations. Let x represent a new feasible active point added to the list L_k, at iteration k. Then, $\min_{y \in L_k}(\|x - y\| - r_y) > 0$ or there should have been an active point $y \in L_k$ such that $\|x - y\| - r_y \leq 0$ and $f(x) < f(y) - \rho(\alpha_y)$.

Let us assume that for each successful iteration k there is always a new active point, $x_{k+1} \in \Omega$, to be added to L_k, such that $\min_{y \in L_k}(\|x_{k+1} - y\| - r_y) > 0$. Thus,

$$\forall y \in L_k, \|x_{k+1} - y\| > r_y \geq d_{max}\alpha_y > d_{max}\alpha_* > 0.$$

Once a point is added to the point list it will always remain in it (possibly becoming inactive). Thus, at each successful iteration the measure of

$$\Omega \setminus \bigcup_{k \in S} B(x_{k+1}; d_{max}\alpha_*)$$

decreases by a strictly positive quantity. Here S represents the set of indexes corresponding to successful iterations and $B(x_{k+1}; d_{max}\alpha_*)$ the open ball of radius $d_{max}\alpha_*$, centered at x_{k+1}. Since Ω is bounded there should have been $k^* \in \mathbb{N}$ such that for

each successful iteration $k \geq k^*$ and for each new feasible active point x added to L_k, there is an active point $y \in L_k$, such that $\|x-y\| - r_y \leq 0$ and $f(x) < f(y) - \rho(\alpha_y) \leq f(y) - \rho(\alpha_*)$, which changes the corresponding status to inactive.

Points are only added to the list at successful or merging iterations. For each point x added to L_k at a merging iteration, there is also an active point $y \in L_k$, which changes its status to inactive, and such that the previous inequalities apply:

$$\|x-y\| - r_y \leq 0 \text{ and } f(x) < f(y) - \rho(\alpha_y) \leq f(y) - \rho(\alpha_*).$$

Thus, for each successful or merging iteration $k \geq k^*$, each time a point is added to the list, the value of f would decrease by at least $\rho(\alpha_*) > 0$, for at least one active point $y \in L_k$. After a finite number of iterations, there should have been an effective decrease of at least $\rho(\alpha_*) > 0$ in the value of the objective function. Since there is an infinite number of successful iterations, this contradicts Assumption 3.2. □

3.3. Refining subsequences, refining directions and generalized directional derivatives. The convergence analysis of GLODS will rely on its behavior in the so-called *refining subsequences*. This particular type of subsequences of unsuccessful iterations was first defined by Audet and Dennis [4] in the context of GPS.

DEFINITION 3.4. *A subsequence $\{x_k\}_{k\in K}$ of iterates corresponding to unsuccessful poll steps is said to be a refining subsequence if $\{\alpha_k\}_{k\in K}$ converges to zero.*

The existence of at least one convergent refining subsequence is a direct consequence of Theorems 3.2 and 3.3, jointly with Assumptions 3.1 and 3.2 (see, for example [10]).

THEOREM 3.5. *Let the conditions required for establishing Theorem 3.2 or Theorem 3.3 hold, additionally to Assumption 3.1 and, when $\bar{\rho}(.)$ is a forcing function, Assumption 3.2. Algorithm 2.1 generates at least one refining subsequence $\{x_k\}_{k\in K}$, converging to $x_* \in \Omega$.*

GLODS behavior will be analysed in limit points of convergent refining subsequences, along *refining directions* (again, a concept introduced by Audet and Dennis [6]).

DEFINITION 3.6. *Let x_* be the limit point of a convergent refining subsequence $\{x_k\}_{k\in K}$. If the limit $\lim_{k\in K'} d_k/\|d_k\|$ exists, where $K' \subseteq K$ and $d_k \in D_k$, and if $x_k + \alpha_k d_k \in \Omega$, for sufficiently large $k \in K'$, then this limit is said to be a refining direction for x_*.*

Refining directions exist trivially in the unconstrained case $\Omega = \mathbb{R}^n$.

Since GLODS is intended for the minimization of nonsmooth functions, a possible stationarity result would consist in establishing the nonnegativity of the Clarke-Jahn generalized directional derivatives [18], computed for a limit point of the sequence of iterates generated by the algorithm, for the whole set of directions belonging to the Clarke generalized tangent cone to the feasible region [9].

DEFINITION 3.7. *Let f be Lipschitz continuous near a point $x_* \in \Omega$. We say that x_* is a Clarke critical point of f in Ω if, for all directions $d \in T_\Omega^{Cl}(x_*)$, $f^\circ(x_*; d) \geq 0$.*

Let us start by providing the appropriate definitions, namely of Clarke tangent cone to the feasible region [9], $T_\Omega^{Cl}(x_*)$, and of Clarke-Jahn generalized directional derivatives [18], $f^\circ(x_*; d)$.

DEFINITION 3.8. *A vector $d \in \mathbb{R}^n$ is said to be a Clarke tangent vector to the set $\Omega \subset \mathbb{R}^n$ at the point x in the closure of Ω if for every sequence $\{y_k\}$ of elements of Ω that converges to x and for every sequence of positive real numbers $\{t_k\}$ converging to zero, there exists a sequence of vectors $\{w_k\}$ converging to d such that $y_k + t_k w_k \in \Omega$.*

The set $T_\Omega^{Cl}(x)$ of all Clarke tangent vectors to Ω at x is called the Clarke tangent cone to Ω at x, and it is a generalization of the tangent cone commonly used in Nonlinear Programming (see, e.g., [30, Definition 12.2 and Figure 12.8]).

The interior of the Clarke tangent cone defines the hypertangent cone, denoted by $T_\Omega^H(x)$, which is formed by the set of all the hypertangent vectors to Ω at x.

DEFINITION 3.9. *A vector $d \in \mathbb{R}^n$ is said to be a hypertangent vector to the set $\Omega \subset \mathbb{R}^n$ at the point x in Ω if there exists a scalar $\epsilon > 0$ such that*

$$y + tw \in \Omega, \quad \forall y \in \Omega \cap B(x;\epsilon), \quad w \in B(d;\epsilon), \quad \text{and} \quad 0 < t < \epsilon.$$

Conversely, the Clarke tangent cone is the closure of the hypertangent cone.

Assuming the Lipschitz continuity of f near x, the Clarke-Jahn generalized derivatives can be defined along directions d in the hypertangent cone to Ω at x,

$$f^\circ(x;d) \;=\; \limsup_{\substack{x' \to x,\, x' \in \Omega \\ t \downarrow 0,\, x' + td \in \Omega}} \frac{f(x' + td) - f(x')}{t}.$$

Considering limits, this definition was extended by Audet and Dennis [6, Proposition 3.9] to directions v belonging to the tangent cone to Ω at x, but not in the hypertangent cone:

$$f^\circ(x;v) = \lim_{d \in T_\Omega^H(x),\, d \to v} f^\circ(x;d).$$

If the objective function f is strictly differentiable at x_*, meaning that the Clarke generalized gradient is a singleton, namely $\nabla f(x_*)$, then the previous definition of Clarke stationarity (Definition 3.7) can be restated using the gradient vector.

DEFINITION 3.10. *Let f be strictly differentiable at a point $x_* \in \Omega$. We say that x_* is a Clarke-KKT critical point of f in Ω if, for all directions $d \in T_\Omega^{Cl}(x_*)$, $\nabla f(x_*)^\top d \geq 0$.*

3.4. Convergence results. For obtaining the desired convergence result, we start by establishing the nonnegativity of the Clarke-Jahn generalized directional derivatives, computed at the limit point of a convergent refining subsequence, along refining directions.

THEOREM 3.11. *Consider a refining subsequence $\{x_k\}_{k \in K}$ converging to $x_* \in \Omega$ and let $d \in T_\Omega^H(x_*)$ be a refining direction for x_*. Assume that f is Lipschitz continuous near x_*. Then $f^\circ(x_*;d) \geq 0$.*

Proof. Let $\{x_k\}_{k \in K}$ be a refining subsequence converging to $x_* \in \Omega$ and

$$d = \lim_{k \in K''} d_k / \|d_k\| \in T_\Omega^H(x_*)$$

a refining direction for x_*, with $d_k \in D_k$ and $x_k + \alpha_k d_k \in \Omega$, $\forall k \in K'' \subseteq K$.

Since f is Lipchitz continuous near x_*, the Clarke-Jahn generalized directional

derivative is well defined for x_* and we have:

$$\begin{aligned} f^\circ(x_*;d) = & \limsup_{\substack{x \to x_*, x \in \Omega \\ t \downarrow 0, x+td \in \Omega}} \frac{f(x+td)-f(x)}{t} \\ & \geq \limsup_{k \in K''} \frac{f(x_k + \alpha_k \|d_k\|(d_k/\|d_k\|)) - f(x_k)}{\alpha_k \|d_k\|} + \\ & \quad + \frac{f(x_k + \alpha_k \|d_k\| d) - f(x_k + \alpha_k \|d_k\|(d_k/\|d_k\|))}{\alpha_k \|d_k\|} \\ & = \limsup_{k \in K''} \frac{f(x_k + \alpha_k \|d_k\|(d_k/\|d_k\|)) - f(x_k)}{\alpha_k \|d_k\|} + e_k \\ & = \limsup_{k \in K''} \frac{f(x_k + \alpha_k d_k) - f(x_k) + \bar{\rho}(\alpha_k)}{\alpha_k \|d_k\|} - \frac{\bar{\rho}(\alpha_k)}{\alpha_k \|d_k\|} + e_k. \end{aligned}$$

The first inequality follows from $\{x_k\}_{k \in K''}$ being a feasible refining subsequence and the fact that $x_k + \alpha_k d_k$ is feasible for $k \in K''$. The term $|e_k|$ is bounded above by $\nu||d - d_k/\|d_k\|\|$, where ν is the Lipschitz constant of f near x_*. Also, note that the limit $\lim_{k \in K''} \bar{\rho}(\alpha_k)/(\alpha_k \|d_k\|)$ is 0 for both globalization strategies (Subsections 3.1 and 3.2). In the case of using integer lattices (Subsection 3.1), one uses $\bar{\rho}(\cdot) \equiv 0$. When imposing sufficient decrease (Subsection 3.2), this limit follows from the properties of the forcing function and the existence of d_{min}, a strictly positive lower bound for the norm of the poll directions.

Since $x_k + \alpha_k d_k$ corresponds to a point evaluated at the unsuccessful iteration k, it was necessarily compared with the active poll center x_k. Thus

$$f(x_k + \alpha_k d_k) \geq f(x_k) - \bar{\rho}(\alpha_k) \Leftrightarrow f(x_k + \alpha_k d_k) - f(x_k) + \bar{\rho}(\alpha_k) \geq 0.$$

The previous statements allow us to conclude that $f^\circ(x_*;d) \geq 0$. □

If we assume strict differentiability of f at the point x_*, the conclusion of the above result will be $\nabla f(x_*)^\top d \geq 0$.

COROLLARY 3.1. *Consider a refining subsequence $\{x_k\}_{k \in K}$ converging to $x_* \in \Omega$ and let $d \in T_\Omega^H(x_*)$ be a refining direction for x_*. Assume that f is strictly differentiable at x_*. Then $\nabla f(x_*)^\top d \geq 0$.*

By assuming density of the set of refining directions associated with x_*, the previous results can be extended to the whole set of directions belonging to the Clarke tangent cone.

THEOREM 3.12. *Consider a refining subsequence $\{x_k\}_{k \in K}$ converging to $x_* \in \Omega$. Assume that $T_\Omega^{Cl}(x_*) \neq \emptyset$. If f is Lipschitz continuous near x_* and the set of refining directions for x_* is dense in $T_\Omega^{Cl}(x_*)$, then x_* is a Clarke critical point.*

In addition, if f is strictly differentiable at x_*, then this point is a Clarke-KKT critical point.

Proof. Let $v \in T_\Omega^{Cl}(x_*)$. Since the set of refining directions for x_* is dense in $T_\Omega^{Cl}(x_*)$ then $v = \lim_{r \in R} d_r$ with d_r a refining direction for x_* belonging to $T_\Omega^H(x_*)$. Thus (see [6])

$$f^\circ(x_*;v) = \lim_{\substack{d \to v \\ d \in T_\Omega^H(x_*)}} f^\circ(x_*;d) = \lim_{r \in R} f^\circ(x_*;d_r) \geq 0.$$

The second statement of the theorem results trivially. □

Problem	n	l	u	loc	$glob$
ackley [1]	10	-30	30	–	1
aluffi_pentini [1]	2	-10	10	2	1
becker_lago [1]	2	-10	10	4	4
bohachevsky [1]	2	-50	50	–	1
branin_hoo [31, 17]	2	$[-5\ 0]^\top$	$[10\ 15]^\top$	3	3
cauchy [8]	4	3	17	–	–
cauchy [8]	10	2	26	–	–
cosine_mixture [8]	2	-1	1	–	–
cosine_mixture [8]	4	-1	1	–	–
dekkers_aarts [1]	2	-20	20	3	2
epistatic_michalewicz [1]	5	0	π	–	1
epistatic_michalewicz [1]	10	0	π	–	1
exponencial [8]	2	-1	1	–	1
exponencial [8]	4	-1	1	–	1
fifteenn_local_minima [8]	2	-10	10	15^2	1
fifteenn_local_minima [8]	4	-10	10	15^4	1
fifteenn_local_minima [8]	6	-10	10	15^6	1
fifteenn_local_minima [8]	8	-10	10	15^8	1
fifteenn_local_minima [8]	10	-10	10	15^{10}	1
goldstein_price [8]	2	-2	2	4	1
griewank [36, 16]	5	-600	600	–	1
griewank [36]	10	-400	400	–	1
gulf [1]	3	$[0.1\ 0\ 0]^\top$	$[100\ 25.6\ 5]^\top$	–	1
hartman_4 [8]	3	0	1	4	1
hartman_4 [8]	6	0	1	4	1
hosaki [1]	2	$[0\ 0]^\top$	$[5\ 6]^\top$	2	1
kowalik [1]	4	0	0.42	–	1
langerman [1]	10	0	10	–	1
mccormick [1]	2	$[-1.5\ -3]^\top$	$[4\ 3]^\top$	2	1
miele_cantrell [8]	4	-10	10	–	1

TABLE 4.1

A description of the test set (first part). The variable n represents the dimension of the problem, l and u, lower and upper bounds, respectively, on the problem variables, loc and glob, the number of local and global minimizers, respectively, reported in the literature.

4. Numerical experiments. A basic version of the proposed algorithm was implemented in MATLAB, and numerically tested in a set comprising 61 bound constrained minimization problems collected from the global optimization literature, with a number of variables between 2 and 10. Both the numerical implementation and the test set considered are freely available (under a GNU Lesser General Public License) at:

`http://ferrari.dmat.fct.unl.pt/personal/alcustodio/GLODS`.

Tables 4.1 and 4.2 provide additional information concerning the test set, namely the number of problem variables (n), the lower (l) and upper (u) bounds suggested in the literature, and the total number of known local (loc) and global ($glob$) minimizers.

A careful analysis of these tables allows us to conclude that the majority of the problems presents a symmetric feasible region. In a considerable number of problems, the global minimum will be located in the center of it. Since the inclusion of points

Problem	n	l	u	loc	$glob$
multi_gaussian [1]	2	-2	2	5	1
neumaier2 [1]	4	0	4	–	1
neumaier3 [1]	10	-100	100	–	1
paviani [1]	10	2.001	9.999	–	1
periodic [1]	2	-10	10	50	1
poissonian [8]	2	$[1\ 1]^\top$	$[21\ 8]^\top$	–	–
powell [1]	4	-10	10	1	1
rastrigin [1]	10	-5.12	5.12	–	1
rosenbrock [8, 17]	2	-5.12	5.12	1	1
salomon [1]	5	-100	100	–	1
salomon [1]	10	-100	100	–	1
schaffer1 [1]	2	-100	100	–	1
schaffer2 [1]	2	-100	100	–	1
schwefel [1]	10	-500	500	–	1
shekel_45 [8]	4	0	10	5	1
shekel_47 [8]	4	0	10	7	1
shekel_410 [8]	4	0	10	10	1
shekel_foxholes [1]	5	0	10	–	1
shekel_foxholes [1]	10	0	10	–	1
shubert [1]	2	-10	10	760	18
sinusoidal [1]	10	0	180	–	1
sixhumpcamel [8, 17]	2	$[-3\ -2]^\top$	$[3\ 2]^\top$	6	2
sphere [36, 16]	3	-5.12	5.12	1	1
storn tchebychev [1]	9	-128	128	–	1
tenn_local_minima [8]	2	-10	10	10^2	1
tenn_local_minima [8]	4	-10	10	10^4	1
tenn_local_minima [8]	6	-10	10	10^6	1
tenn_local_minima [8]	8	-10	10	10^8	1
three_hump_camel [1]	2	-5	5	3	1
transistor [1]	9	-10	10	–	1
wood [8]	4	-10	10	–	1

TABLE 4.2

A description of the test set (second part). The variable n represents the dimension of the problem, l and u, lower and upper bounds, respectively, on the problem variables, loc and glob, the number of local and global minimizers, respectively, reported in the literature.

close to the center of the feasible region is a common practice for initializing solvers, we decided to perturb the bounds considered for each problem. The modified global optimization problems would then be defined as:

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & x \in \Omega \equiv [l, u - 0.35(u-l)] \subset \mathbb{R}^n, \end{aligned}$$

where $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is a real-extended value function and $l, u \in \mathbb{R}^n$ are the ones reported in Tables 4.1 and 4.2.

GLODS is initialized with n points, evenly spaced in a line joining the lower and upper bounds of the modified problems. Additionally, the center of the feasible region is also used for initialization. The algorithm performs opportunistic polling,

considering the positive basis $[I \quad -I]$, where I denotes the identity matrix. Before selecting a new poll center, the list of feasible active points is ordered in a greedy fashion: points not yet identified as local minimizers (meaning that the corresponding step size parameter equals or exceeds 10^{-8}), corresponding to the lower objective function values would be moved to the top. Poll centers would be selected from the active points in the top of the ordered list. Globalization is based in integer lattices, like described in Section 3.1. The step size parameter and the comparison radius were initialized as 1, being the step size halved at unsuccessful iterations and doubled at successful ones. The algorithm would stop when a maximum of 20000 function evaluations is reached or when all the values for the step size parameter corresponding to active points are smaller than 10^{-8}.

As comparison tool we used data profiles, proposed by Moré and Wild [28] for accessing the performance of derivative-free optimization solvers when there are constraints on the computational budget. Let $\mathcal{S}$ and $\mathcal{P}$ represent the set of solvers and the set of problems to be tested, respectively. In a data profile, the numerical performance of each solver is represented by a curve, where each pair $(\sigma, d_s(\sigma))$ represents the percentage of problems, $d_s(\sigma)$, solved by algorithm $s \in \mathcal{S}$ for an equivalent budget of σ simplex gradient estimates (recall that for a problem of dimension n_p, $n_p + 1$ function evaluations are required to compute such a simplex). If $h_{p,s}$ represents the number of function evaluations required for algorithm $s \in \mathcal{S}$ to solve problem $p \in \mathcal{P}$ (up to a certain accuracy), the data profile cumulative function is given by

$$d_s(\sigma) \;=\; \frac{1}{|\mathcal{P}|}\left|\left\{p \in \mathcal{P} : \frac{h_{p,s}}{n_p+1} \leq \sigma\right\}\right|. \tag{4.1}$$

With this purpose, a problem is considered to be solved to an accuracy level τ if the decrease obtained from the initial objective function value $(f(x_0) - f(x))$ is at least $1-\tau$ of the best decrease obtained for all the solvers considered $(f(x_0)-f_L)$, meaning:

$$f(x_0) - f(x) \;\geq\; (1-\tau)[f(x_0) - f_L].$$

In the numerical experiments reported, the accuracy level was set equal to 10^{-5}.

4.1. Comparing different versions of GLODS. The flexibility inherent to the description of Algorithm 2.1 allows us to regard GLODS as a general class of global derivative-free optimization methods, comprising several instances depending, for example, on the definition of the search step. In the present numerical experiments, different strategies were considered for its definition, namely:

- random sampling [35];
- Latin hypercube sampling [27];
- Sobol sequences [22];
- Halton sequences [22];
- 2^n-Centers.

For the first four strategies, each time the search step was performed, 2^n points were generated. Strategy 2^n-Centers was implemented as described in Section 2.1.

As mentioned before, the search step is responsible for the initialization of new direct searches, allowing to explore the whole feasible region and possibly directing search to an area of attraction of the global minimum of the problem. Thus, there is no need to consider it at every iteration. When the current number of active points not yet identified as local minimizers (meaning that the corresponding step size parameter equals or exceeds 10^{-8}) equals one, the search step will be performed,

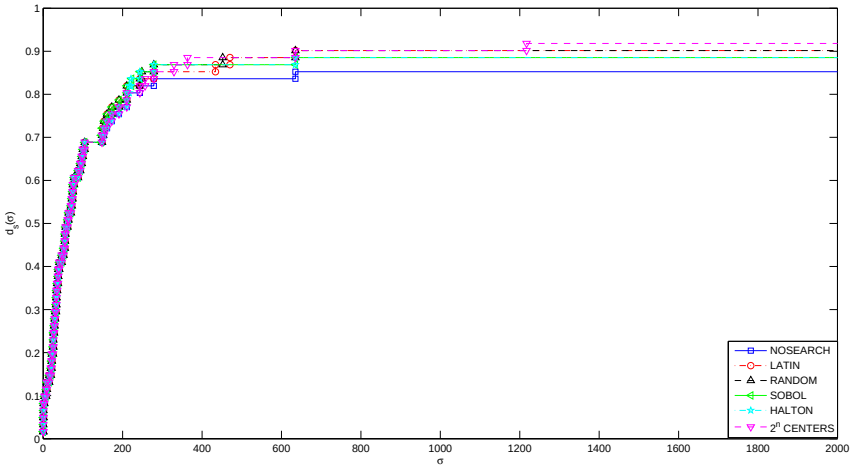

FIG. 4.1. *Comparing different strategies for defining the search step in GLODS (worst runs).*

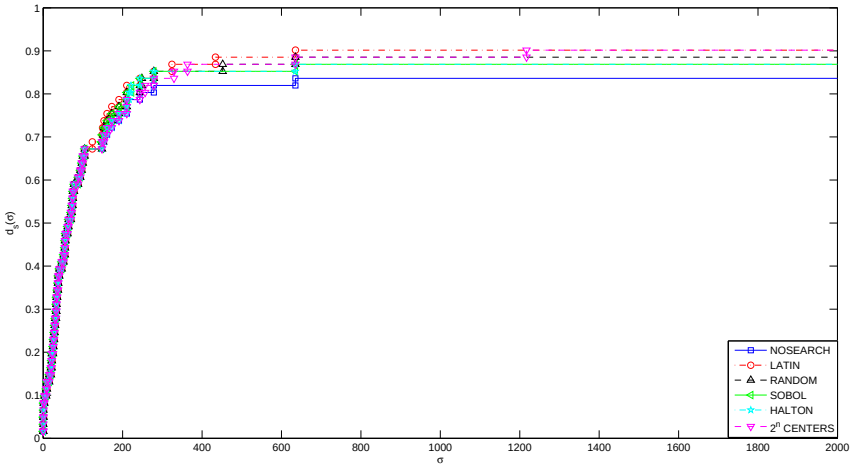

FIG. 4.2. *Comparing different strategies for defining the search step in GLODS (best runs).*

possibly initializing new direct searches in regions not yet explored. Other criteria could have been implemented, again resulting in different algorithmic instances of GLODS class.

Figures 4.1 and 4.2 report the results obtained when considering different types of search steps. For strategies with random behavior, namely when the definition of the search step is based in random or Latin hypercube sampling, 10 runs were performed for each problem and the best and worst runs were selected. The best run was defined as the one that achieves first the best final value. The worst run corresponded to the one that achieves last the worst final value.

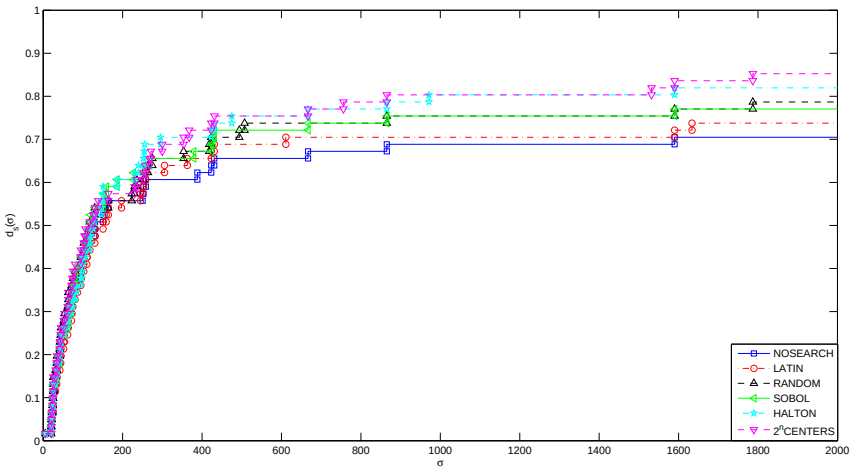

FIG. 4.3. *Comparing different strategies for defining the search step in GLODS, with random initialization (worst runs).*

From the analysis of the profiles, we can say that a search step defined using 2^n-Centers provides a slight advantage over the remaining strategies. Nevertheless, the corresponding gain is small, even when comparing it with a version which does not consider any search step. These numerical results could be related with the initialization of the algorithm (line sampling). Recall that we have perturbed the upper bound of each problem, but that possibly was not enough to avoid the proximity of the initialization from the global minimizers.

To better assess the impact of the implementation of different strategies in the search step of the algorithm, the experiments were repeated, considering an initialization based in n randomly generated points. The corresponding data profiles are reported in Figures 4.3 and 4.4.

The advantage of defining a search step is now clearer. Strategies Halton and 2^n-Centers positively distinguish from the remaining ones. The remaining deterministic strategy, based in the use of Sobol sequences, also presents a good performance. Thus, in what follows, we will consider GLODS with three possibilities for defining the search step: 2^n-Centers, Halton, and Sobol.

4.2. Assessing performance with other solvers. GLODS selected versions were compared against two other global derivative-free optimization solvers, namely:

1. DIRECT – DIviding RECTangles [12];
2. MCS – Global optimization by Multilevel Coordinate Search [16];

Both solvers proceed by dividing the feasible region in subdomains. The partitioning is not uniform, and preference is given to regions not yet explored or to the most promising ones, considering the corresponding objective functions values. In the case of MCS, minimization of quadratic polynomial models is additionally used to improve the local searches.

Solvers were run with default values, with exception to the maximum number of function evaluations allowed, which was set equal to 20000. As mentioned before, GLODS would also stop if the maximum value for the step size parameter correspon-

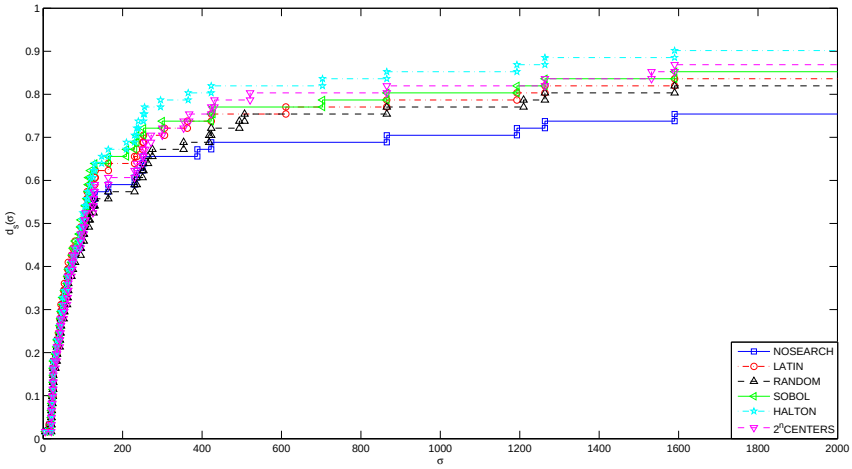

FIG. 4.4. *Comparing different strategies for defining the search step in GLODS, with random initialization (best runs).*

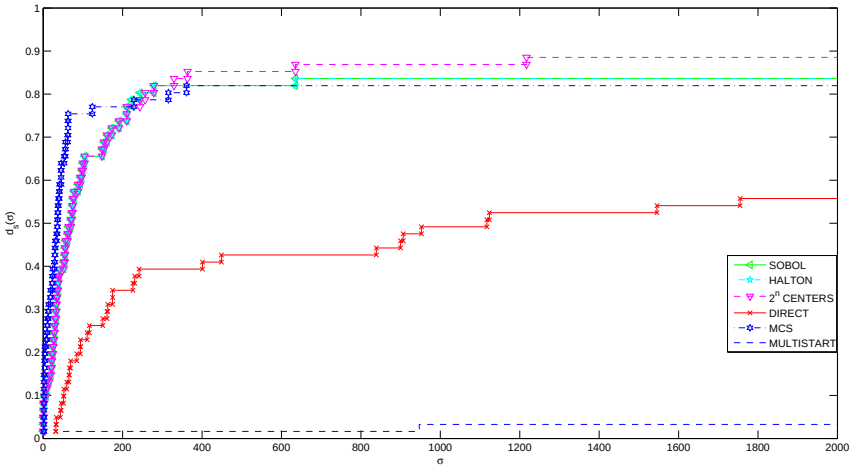

FIG. 4.5. *Comparing GLODS with other solvers.*

ding to active points is smaller than 10^{-8}.

Additionally, a basic implementation of a pure multistart directional direct-search method was considered. Direct searches are initialized with random sampling and coordinated search is used for local optimization. Again, a maximum budget of 20000 functions evaluations was considered and each local search would be ended when the corresponding step size parameter was smaller than 10^{-8}.

Results are reported in Figure 4.5. For multistart, we have selected the best run from a total of 10 that were performed.

The bad performance of the pure multistart strategy is not surprising. It is clear the advantage of MCS and GLODS versions over DIRECT. For small budgets of functions evaluations, for the test set and the level of accuracy considered, MCS presents the best performance. Nevertheless, GLODS can be considered competitive, presenting the best performance for budgets of moderate size.

Detailed numerical results for MCS, DIRECT and the three versions of GLODS can be found in Appendix A, in Tables A.1 and A.2.

4.3. Global and local minimizers. Differently from MCS and DIRECT, GLODS intends not only to compute the global minimum of a given problem, but also to provide the corresponding different global and local minimizers. In an attempt to evaluate the ability of GLODS to reach the proposed goal, we have returned to the original unperturbed global minimization problem:

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & x \in \Omega \equiv [l, u] \subset \mathbb{R}^n. \end{aligned}$$

This formulation would allow us to use the minimum global values and the local and global minimizers reported in the global optimization literature. Nevertheless, only for 13 of the 61 problems previously considered we could get the complete information. For each problem, Tables 4.3, 4.4, and 4.5 report the minimum global value found in the literature (Global Minimum Reported), the best function value ($fvalue$) computed by each solver, and the number of function evaluations required to compute it ($fevals$). Problems periodic, powell, and sphere correspond to the situation where the initialization provided is close to the global minimizer, which motivated the use of a perturbed feasible region.

		Global Minimum	2^n-CENTERS		SOBOL	
Problem	n	Reported	$fvalue$	$fevals$	$fvalue$	$fevals$
aluffi_pentini	2	-3.5230E-01	-3.5239E-01	97	-3.5239E-01	97
becker_lago	2	0.0000E+00	0.0000E+00	22	0.0000E+00	22
branin_hoo	2	3.9789E-01	3.9789E-01	228	3.9789E-01	228
dekkers_aarts	2	-2.4777E+04	-2.4777E+04	66	-2.4777E+04	66
mccormick	2	-1.9133E+00	-1.9132E+00	122	-1.9132E+00	122
periodic	2	9.0000E-01	9.0000E-01	2	9.0000E-01	2
powell	4	0.0000E+00	0.0000E+00	4	0.0000E+00	4
rosenbrock	2	0.0000E+00	2.1510E-24	9107	2.1510E-24	9107
shekel_45	4	-1.0153E+01	-1.0153E+01	1956	-1.0153E+01	1956
shekel_47	4	-1.0403E+01	-5.1288E+00	3294	-5.0877E+00	2568
shekel_410	4	-1.0536E+01	-5.1756E+00	4099	-5.1285E+00	2452
sixhumpcamel	2	-1.0316E+00	-1.0316E+00	147	-1.0316E+00	147
sphere	3	0.0000E+00	0.0000E+00	1	0.0000E+00	1

TABLE 4.3

Numerical results for the unperturbed problems (first part). The variable $fvalue$ represents the minimum objective function value computed by the corresponding solver, and $fevals$, the number of function evaluations required to achieve it.

With the exception in problems shekel_47 and shekel_410, where GLODS were unable to compute the global minimum value, all the final solutions computed by the different solvers present a quality similar to the global minima reported in the literature. Although, there are differences in what concerns the number of function

Problem	n	Global Minimum Reported	HALTON $fvalue$	HALTON $fevals$	MCS $fvalue$	MCS $fevals$
aluffi_pentini	2	-3.5230E-01	-3.5239E-01	97	-3.5239E-01	151
becker_lago	2	0.0000E+00	0.0000E+00	22	0.0000E+00	22
branin_hoo	2	3.9789E-01	3.9789E-01	228	3.9789E-01	80
dekkers_aarts	2	-2.4777E+04	-2.4777E+04	66	-2.4777E+04	140
mccormick	2	-1.9133E+00	-1.9132E+00	122	-1.9132E+00	73
periodic	2	9.0000E-01	9.0000E-01	2	9.0000E-01	4
powell	4	0.0000E+00	0.0000E+00	4	0.0000E+00	8
rosenbrock	2	0.0000E+00	2.1510E-24	9107	5.7606E-21	342
shekel_45	4	-1.0153E+01	-1.0153E+01	1956	-1.0153E+01	159
shekel_47	4	-1.0403E+01	-5.1288E+00	3404	-1.0403E+01	224
shekel_410	4	-1.0536E+01	-5.1756E+00	3325	-1.0536E+01	189
sixhumpcamel	2	-1.0316E+00	-1.0316E+00	147	-1.0316E+00	103
sphere	3	0.0000E+00	0.0000E+00	1	0.0000E+00	6

TABLE 4.4

Numerical results for the unperturbed problems (second part). The variable $fvalue$ represents the minimum objective function value computed by the corresponding solver, and $fevals$, the number of function evaluations required to achieve it.

Problem	n	Global Minimum Reported	DIRECT $fvalue$	DIRECT $fevals$
aluffi_pentini	2	-3.5230E-01	-3.5239E-01	19614
becker_lago	2	0.0000E+00	2.5386E-19	23056
branin_hoo	2	3.9789E-01	3.9789E-01	8728
dekkers_aarts	2	-2.4777E+04	-2.4777E+04	1120
mccormick	2	-1.9133E+00	-1.9132E+00	916
periodic	2	9.0000E-01	9.0000E-01	1
powell	4	0.0000E+00	0.0000E+00	1
rosenbrock	2	0.0000E+00	9.9944E-23	20344
shekel_45	4	-1.0153E+01	-1.0153E+01	254
shekel_47	4	-1.0403E+01	-1.0403E+01	4878
shekel_410	4	-1.0536E+01	-1.0536E+01	4938
sixhumpcamel	2	-1.0316E+00	-1.0316E+00	3008
sphere	3	0.0000E+00	0.0000E+00	1

TABLE 4.5

Numerical results for the unperturbed problems (third part). The variable $fvalue$ represents the minimum objective function value computed by the corresponding solver, and $fevals$, the number of function evaluations required to achieve it.

evaluations required. In this last criterion, GLODS occupies a medium position, being DIRECT the solver presenting the worst performance for the test set considered. We recall that GLODS attempts to compute all the local minimizers of each problem, which surely contributes for the observed performance. Table 4.6 records the number of local minimizers reported in the literature and the final number of active points computed by the different deterministic variants of GLODS. When the code stops based on the value of the step size parameter, active points indicate local minimizers.

For problem rosenbrock, when the search step is based in the use of Halton sequences, GLODS ends the computation with four active points, but only one of them presents a step size parameter lower than 10^{-8}. In fact, in this situation, if we disable

Problem	n	Number of Local Min	2^n-CENTERS Local Min	SOBOL Local Min	HALTON Local Min
aluffi_pentini	2	2	1	1	1
becker_lago	2	4	4	2	3
branin_hoo	2	3	2	2	3
dekkers_aarts	2	3	1	2	2
mccormick	2	2	2	2	2
periodic	2	50	6	8	6
powell	4	1	1	1	1
rosenbrock	2	1	1	1	4
shekel_45	4	5	4	4	5
shekel_47	4	7	4	4	6
shekel_410	4	10	5	3	7
sixhumpcamel	2	6	2	2	2
sphere	3	1	1	1	1

TABLE 4.6
Number of local minimizers.

the stopping criterion based in the budget of 20000 function evaluations, the code will end with only one active point corresponding to the global minimizer.

5. Conclusions. We have proposed a new class of algorithms, based in directional direct search coupled with multistart strategies, to solve global derivative-free optimization problems. The key idea, which could be imported to other algorithmic frameworks, is to explore the whole feasible region by initializing local searches from different feasible points, but avoiding unnecessary computations by merging local searches which are considered to be sufficiently close. A measure of closeness is given by a comparison radius, directly related to the step size parameter.

Assuming locally Lipschitz continuity of the objective function defining the optimization problem, the convergence of the algorithm was analysed. Results could be further generalized for discontinuous objective functions following the steps in [39].

A basic numerical implementation of GLODS algorithm has shown to be competitive with well-established global derivative-free optimization solvers, for a set of global optimization problems. This numerical implementation presents the additional feature of allowing to compute not only the global minimum, but different local minimizers. There are several ways in which the numerical efficiency of the considered implementation could be improved. In particular, the general structure of directional direct search and multistart strategies strongly suggest that benefits could result from parallelization.

REFERENCES

[1] M. M. Ali, C. Khompatraporn, and Z. B. Zabinsky. A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems. *J. Global Optim.*, 31:635–672, 2005.

[2] I. Andricioaei and J. E. Straub. Global optimization using bad derivatives: Derivative-free method for molecular energy minimization. *J. Computational Chemistry*, 19:1445–1455, 1998.

[3] C. Audet, V. Béchard, and S. Le Digabel. Nonsmooth optimization through mesh adaptive direct search and variable neighborhood search. *J. Global Optim.*, 41:299–318, 2008.

[4] C. Audet and J. E. Dennis Jr. Analysis of generalized pattern searches. *SIAM J. Optim.*, 13:889–903, 2003.

[5] C. Audet and J. E. Dennis Jr. A pattern search filter method for nonlinear programming without derivatives. *SIAM J. Optim.*, 14:980–1010, 2004.

[6] C. Audet and J. E. Dennis Jr. Mesh adaptive direct search algorithms for constrained optimization. *SIAM J. Optim.*, 17:188–217, 2006.

[7] C. Audet and J. E. Dennis Jr. A progressive barrier for derivative-free nonlinear programming. *SIAM J. Optim.*, 20:445–472, 2009.

[8] P. Brachetti, M. F. Ciccoli, G. Di Pillo, and S. Lucidi. A new version of the Price's algorithm for global optimization. *J. Global Optim.*, 10:165–184, 1997.

[9] F. H. Clarke. *Optimization and Nonsmooth Analysis*. John Wiley & Sons, New York, 1983. Reissued by SIAM, Philadelphia, 1990.

[10] A. R. Conn, K. Scheinberg, and L. N. Vicente. *Introduction to Derivative-Free Optimization*. MPS-SIAM Series on Optimization. SIAM, Philadelphia, 2009.

[11] C. Davis. Theory of positive linear dependence. *Amer. J. Math.*, 76:733–746, 1954.

[12] D. E. Finkel. *DIRECT Optimization Algorithm User Guide*, 2003. http://www4.ncsu.edu/definkel/research/index.html.

[13] W. Gao and C. Mi. Hybrid vehicle design using global optimisation algorithms. *Int. J. Electric and Hybrid Vehicles*, 1:57–70, 2007.

[14] A.-R. Hedar and M. Fukushima. Hybrid simulated annealing and direct search method for nonlinear unconstrained global optimization. *Optim. Methods Softw.*, 17:891–912, 2002.

[15] A.-R. Hedar and M. Fukushima. Tabu Search directed by direct search methods for nonlinear global optimization. *European J. Oper. Res.*, 170:329–349, 2006.

[16] W. Huyer and A. Neumaier. Global optimization by multilevel coordinate search. *J. Global Optim.*, 14:331–355, 1999.

[17] W. Huyer and A. Neumaier. SNOBFIT– Stable noisy optimization by branch and fit. *ACM Trans. Math. Software*, 35:9:1–9:25, 2008.

[18] J. Jahn. *Introduction to the Theory of Nonlinear Optimization*. Springer-Verlag, Berlin, 1996.

[19] D. Jones, C. Perttunen, and B. Stuckman. Lipschitzian optimization without the Lipschitz constant. *J. Optim. Theory Appl.*, 79:157–181, 1993.

[20] A. H. G. Rinnooy Kan and G. T. Timmer. Stochastic global optimization methods – Part I: Clustering methods. *Math. Program.*, 39:27–56, 1987.

[21] A. H. G. Rinnooy Kan and G. T. Timmer. Stochastic global optimization methods – Part II: Multi level methods. *Math. Program.*, 39:57–78, 1987.

[22] L. Kocis and W. J. Whiten. Computational investigations of low-discrepancy sequences. *ACM Trans. Math. Software*, 23:266–294, 1997.

[23] T. G. Kolda, R. M. Lewis, and V. Torczon. Optimization by direct search: New perspectives on some classical and modern methods. *SIAM Rev.*, 45:385–482, 2003.

[24] D. Lee, J.-W. Kim, C.-G. Lee, and S.-Y. Jung. Variable mesh adaptive direct search algorithm applied for optimal design of electric machines based on FEA. *IEEE Transactions on Magnetics*, 47:3232–3235, 2011.

[25] M. Leonetti, P. Kormushev, and S. Sagratella. Combining local and global direct derivative-free optimization for reinforcement learning. *Cybernetics and Information Technologies*, 12:53–65, 2012.

[26] M. Locatelli. Relaxing the assumptions of the multilevel single linkage algorithm. *J. Global Optim.*, 13:25–42, 1998.

[27] M. D. McKay, R. J. Beckman, and W. J. Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21:239–245, 1979.

[28] J. J. Moré and S. M. Wild. Benchmarking derivative-free optimization algorithms. *SIAM J. Optim.*, 20:172–191, `http://www.mcs.anl.gov/~more/dfo`, 2009.

[29] R. C. Morgans, C. Q. Howard, A. C. Zander, C. H. Hansen, and D. J. Murphy. Derivative free optimisation in engineering and acoustics. In *14th International Congress on Sound & Vibration*, pages 1–8, 2007.

[30] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer-Verlag, Berlin, second edition, 2006.

[31] U. M. García Palomares. Searching for multiple minima of bound constrained optimization problems using derivative free optimization techniques. In *Proceedings of the Eleventh International Conference on Computational Structures Technology*, page Paper 63, 2012.

[32] U. M. García Palomares, F. J. Gonzalez-Castaño, and J. C. Burguillo-Rial. A combined global & local search (CGLS) approach to global optimization. *J. Global Optim.*, 34:409–426, 2006.

[33] D. Peri, G. Fasano, D. Dessi, and E. F. Campana. Global optimization algorithms in multidisciplinary design optimization. In *2th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, pages 1–12, 2008.

[34] R. G. Regis and C. A. Shoemaker. A quasi-multistart framework for global optimization of expensive functions using response surface models. *J. Global Optim.*, 56:1719–1753, 2013.

[35] T. J. Santner, B. J. Williams, and W. I. Notz. *The Design and Analysis of Computer Experiments*. Springer-Verlag, New York, 2003.

[36] R. Storn and K. Price. Differential Evolution A simple and efficient heuristic for global optimization over continuous spaces. *J. Global Optim.*, 11:341–359, 1997.

[37] V. Torczon. On the convergence of pattern search algorithms. *SIAM J. Optim.*, 7:1–25, 1997.

[38] A. Ismael F. Vaz and L. N. Vicente. A particle swarm pattern search method for bound constrained global optimization. *J. Global Optim.*, 39:197–219, 2007.

[39] L. N. Vicente and A. L. Custódio. Analysis of direct searches for discontinuous functions. *Math. Program.*, 133:299–325, 2012.

Appendix A. Detailed numerical results.

Problem	n	2^n-CENTERS		SOBOL		HALTON		MCS		DIRECT	
		$fvalue$	$fevals$	$fvalue$	$fevals$	$fvalue$	$fevals$	$fvalue$	$fevals$	$fvalue$	$fevals$
ackley	10	2.3842E-09	4386	2.3842E-09	4386	2.3842E-09	4386	3.5388E-01	3590	7.8388E-01	10350
aluffi_pentini	2	-1.5264E-01	167	-1.5264E-01	167	-1.5264E-01	167	-3.5239E-01	158	-3.5239E-01	9410
becker_lago	2	9.9920E-18	315	9.9920E-18	315	9.9920E-18	315	0.0000E+00	21	1.5777E-30	22436
bohachevsky	2	1.9984E-14	299	1.2879E-14	359	1.2879E-14	373	0.0000E+00	30	0.0000E+00	2340
branin_hoo	2	3.9789E-01	226	3.9789E-01	226	3.9789E-01	226	3.9789E-01	73	3.9789E-01	6868
cauchy	4	-2.1994E+01	610	-2.1994E+01	610	-2.1994E+01	610	-2.1994E+01	8	-2.1993E+01	19102
cauchy	10	-6.4588E+01	101	-6.4588E+01	101	-6.4588E+01	101	-6.4588E+01	20	-6.4583E+01	5782
cosine_mixture	2	-2.2000E+00	1	-2.2000E+00	1	-2.2000E+00	1	-2.2000E+00	4	-2.2000E+00	10644
cosine_mixture	4	-4.4000E+00	1	-4.4000E+00	1	-4.4000E+00	1	-4.4000E+00	8	-4.3999E+00	11590
dekkers_aarts	2	-2.4777E+04	342	-2.4777E+04	342	-2.4777E+04	342	-2.4777E+04	123	-2.4777E+04	9282
epistatic_michalewicz	5	-2.7355E+00	492	-3.5049E+00	6422	-2.7355E+00	492	-2.6943E+00	948	-3.7131E+00	10638
epistatic_michalewicz	10	-7.1402E+00	17218	-7.1402E+00	17218	-7.1402E+00	17218	-7.0621E+00	4992	-6.1188E+00	14034
exponencial	2	-1.0000E+00	125	-1.0000E+00	125	-1.0000E+00	125	-1.0000E+00	38	-1.0000E+00	15750
exponencial	4	-1.0000E+00	449	-1.0000E+00	449	-1.0000E+00	449	-1.0000E+00	68	-1.0000E+00	13764
fifteenn_local_minima	2	8.0670E-17	269	8.0670E-17	269	8.0670E-17	269	1.3498E-32	136	1.3498E-32	13518
fifteenn_local_minima	4	8.2446E-17	905	8.2446E-17	905	8.2446E-17	905	1.3498E-32	280	2.8577E-23	22738
fifteenn_local_minima	6	8.4223E-17	1829	8.4223E-17	1829	8.4223E-17	1829	1.3498E-32	438	3.0211E-02	19922
fifteenn_local_minima	8	1.3273E-17	2350	1.3273E-17	2350	1.3273E-17	2350	1.3498E-32	564	7.6973E-02	20254
fifteenn_local_minima	10	1.6261E-16	6021	1.6261E-16	6021	1.6261E-16	6021	1.3498E-32	611	1.1467E-01	21152
goldstein_price	2	3.0000E+00	232	3.0000E+00	232	3.0000E+00	232	3.0000E+00	142	3.0000E+00	2926
griewank	10	6.6584E-02	3726	6.6584E-02	3726	6.6584E-02	3726	8.1197E-02	3552	9.4267E-02	8210
griewank	5	7.0251E-01	772	7.0251E-01	772	7.0251E-01	772	4.9225E-02	1371	1.1086E-01	9860
gulf	3	1.6015E+00	481	1.6015E+00	481	1.6015E+00	481	1.6015E+00	46	1.6015E+00	18584
hartman_4	3	-2.3872E+00	90	-2.3872E+00	90	-2.3872E+00	90	-2.3872E+00	142	-2.3871E+00	542
hartman_4	6	-3.3210E+00	766	-3.3210E+00	766	-3.3210E+00	766	-3.3210E+00	159	-3.3208E+00	1288
hosaki	2	-1.7701E+00	83	-1.7701E+00	83	-1.7701E+00	83	-1.7701E+00	31	-1.7701E+00	4154
kowalik	4	3.0749E-04	19991	3.0749E-04	19991	3.0749E-04	19991	3.0749E-04	225	3.0751E-04	20002
langerman	10	-2.1275E-02	3750	-8.8200E-04	6374	-1.6670E-03	3886	-1.1664E-03	287	-1.3227E-04	17042
mccormick	2	-1.9132E+00	150	-1.9132E+00	150	-1.9132E+00	150	-1.9132E+00	101	-1.9132E+00	15244

TABLE A.1

Numerical results for MCS, DIRECT, and three variants of GLODS (first part). The variable $fvalue$ represents the minimum objective function value computed by the corresponding solver, and $fevals$, the number of function evaluations required to achieve it.

Problem	n	2^n-CENTERS		SOBOL		HALTON		MCS		DIRECT	
		$fvalue$	$fevals$	$fvalue$	$fevals$	$fvalue$	$fevals$	$fvalue$	$fevals$	$fvalue$	$fevals$
miele_cantrell	4	7.5815E-17	20004	7.5815E-17	20004	7.5815E-17	20004	4.7495E-15	806	5.3116E-20	21480
multi_gaussian	2	-1.2970E+00	1173	-1.2168E+00	393	-1.2168E+00	393	-7.4424E-01	32	-1.2970E+00	9536
neumaier2	4	2.3277E+00	401	2.3277E+00	401	2.3277E+00	401	2.3277E+00	116	3.6112E+00	18536
neumaier3	10	-2.1000E+02	6104	-2.1000E+02	6104	-2.1000E+02	6104	-2.1000E+02	336	-2.0999E+02	10332
paviani	10	-1.4053E+01	9	-1.4053E+01	9	-1.4053E+01	9	-1.4053E+01	20	-1.3560E+01	20958
periodic	2	9.0000E-01	827	9.9999E-01	1162	9.9999E-01	833	9.0000E-01	54	9.0000E-01	13184
poissonian	2	7.1252E+01	99	7.1252E+01	99	7.1252E+01	99	7.1252E+01	89	7.1252E+01	2720
powell	4	6.9851E-07	19999	6.9851E-07	19999	6.9851E-07	19999	2.1519E-18	618	2.3904E-05	20048
rastrigin	10	0.0000E+00	4201	0.0000E+00	4201	0.0000E+00	4201	0.0000E+00	205	9.9498E+00	7808
rosenbrock	2	2.5621E-05	19992	2.5621E-05	19992	2.5621E-05	19992	2.0051E-20	108	3.6386E-22	19728
salomon	5	4.9987E-01	339	4.9987E-01	339	4.9987E-01	339	1.0999E+00	402	1.1999E+00	22100
salomon	10	7.9987E-01	13745	1.0999E+00	718	1.0999E+00	718	5.4999E+00	335	2.5298E+00	9808
schaffer1	2	3.7224E-02	479	9.7159E-03	514	9.7159E-03	753	0.0000E+00	32	9.7159E-03	4032
schaffer2	2	2.5098E+00	722	1.2787E-01	708	2.4000E-01	568	3.1349E-02	946	5.4966E-02	9824
schwefel	10	-3.0054E+03	3558	-3.0054E+03	3558	-3.0054E+03	3558	-1.9258E+03	217	-3.2030E+02	21546
shekel_45	4	-1.0153E+01	487	-1.0153E+01	487	-1.0153E+01	487	-1.0153E+01	226	-1.0153E+01	5392
shekel_47	4	-1.0403E+01	430	-1.0403E+01	430	-1.0403E+01	430	-1.0403E+01	186	-1.0403E+01	15138
shekel_410	4	-1.0536E+01	475	-1.0536E+01	475	-1.0536E+01	475	-1.0536E+01	307	-1.0536E+01	13938
shekel_foxholes	5	-2.7010E+00	874	-2.7010E+00	874	-2.7010E+00	874	-8.5795E-01	161	-2.7010E+00	19704
shekel_foxholes	10	-4.4920E-01	4059	-4.4920E-01	4059	-4.4920E-01	4059	-4.4920E-01	309	-4.4822E-01	13374
shubert	2	-4.6511E+01	1348	-1.8673E+02	744	-1.8673E+02	729	-1.8673E+02	1083	-1.8673E+02	10080
sinusoidal	10	-3.1730E+00	9	-3.1730E+00	9	-3.1730E+00	9	-3.1730E+00	20	-3.1724E+00	13156
sixhumpcamel	2	-1.0316E+00	272	-1.0316E+00	272	-1.0316E+00	272	-1.0316E+00	65	-1.0316E+00	3138
sphere	3	1.5716E-17	575	1.5716E-17	575	1.0514E-17	727	0.0000E+00	39	1.9568E-18	21064
storn_tchebychev	9	2.0490E+01	19994	2.0490E+01	19994	2.0490E+01	19994	1.3735E+02	1376	1.8663E+01	20134
tenn_local_minima	2	1.4049E-15	478	1.4049E-15	478	1.4049E-15	478	2.3558E-31	122	2.3558E-31	9112
tenn_local_minima	4	9.7507E-16	1515	9.7507E-16	1515	9.7507E-16	1515	1.1779E-31	183	2.8056E-23	16920
tenn_local_minima	6	4.8689E-16	3011	4.8689E-16	3011	4.8689E-16	3011	7.8527E-32	268	4.9776E-19	21164
tenn_local_minima	8	5.7435E-17	2204	5.7435E-17	2204	5.7435E-17	2204	5.8895E-32	566	4.4223E-14	20262
threehumpcamel	2	1.2768E-17	305	4.6491E-18	350	1.2768E-17	305	1.3535E-39	88	9.2774E-22	22344
transistor	9	1.1095E+03	19998	1.1095E+03	19998	1.1095E+03	19998	1.0088E+02	505	1.8961E+02	11004
wood	4	1.2644E-03	19993	1.2644E-03	19993	1.2644E-03	19993	1.3588E-18	274	7.3673E-03	17772

TABLE A.2

Numerical results for MCS, DIRECT, and three variants of GLODS (second part). The variable $fvalue$ represents the minimum objective function value computed by the corresponding solver, and $fevals$, the number of function evaluations required to achieve it.