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 when the derivatives of the functions defining the problem are not available for use. 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 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 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 a lower-semicontinuous function, defined in a compact set. The lower-semicontinuity property ensures the existence of a global minimum for the function in the feasible region considered, which we try to locate by solving the problem

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

Here $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is a real-extended value function and the compact set $\Omega \subset \mathbb{R}^n$ represents the feasible region. GLODS (Global and Local Optimization using Direct Search) class is designed for computing all the local minimizers of the problem, from which the global minimum would be easily identified.

Solving global optimization problems is a challenging task, with additional difficulties when derivatives are not available for use. Nevertheless, 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 [10] or for reinforcement learning in robotics [18]) to chemistry (molecular conformal optimization problems [2]), acoustics [22] or multidisciplinary design optimization [25].

In derivative-free optimization, stating global convergence of an algorithm means ensuring convergence to a stationary point, regardless of the initial approximation provided to the optimizer (see [7]). When practitioners are faced with the need of

*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/UI0297/2011 (CMA) and the grant PTDC/MAT/116736/2010.

†IDMEC-IST, TU-Lisbon, Av. Rovisco Pais, 1040-001 Lisboa, Portugal and ISEL, Rua Conselheiro Emídio Navarro, 1, 1959-007 Lisboa (jaguilar@dem.ist.utl.pt). Support for this author was provided by ISEL and Fundação para a Ciência e a Tecnologia, Portugal, for the financing of IDMEC-IST in the framework of LAETA.

computing a global minimum, a typical approach is to use a local optimization procedure coupled with a multistart strategy.

Let us consider a problem with a finite number of local minimizers. Pure multistart strategies are generally quite inefficient, since several local searches will converge to the same minimizer. Enhancements to multistart have been previously proposed by other authors, to our knowledge mainly in the context of stochastic algorithms (see for example [14]). 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), will be a clever deterministic alternative to pure multistart. The algorithm will consider a local search procedure based on direct search of directional type [7], 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 [15, 19], 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.

Similarly to other derivative-free optimization algorithms, an extreme barrier approach will be used for the feasible region, 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 $+\infty$.

Section 2 will describe the motivation behind the algorithmic design, also providing a rigorous description of GLODS. Using Clarke nonsmooth analysis [6], Section 3 will establish the convergence properties of GLODS. Numerical experiments in a set of bound constrained global optimization problems are reported in Section 4. 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 minimums, will depend on the search step.

A 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 $\forall d \in D \subseteq \mathcal{D}, 0 < d_{min} \leq \|d\| \leq d_{max}$. 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+1} `= add(`L_k`,`A_k`)` to eventually add some new points in A_k to L_k. If $k = 0$, set $L_0 = L_1$ and go to the poll step. Otherwise, if there is a new active point in L_{k+1} 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 $P_k = \{(x_k + \alpha_k d; \alpha_k; \alpha_k; 0) : d \in D_k \wedge f_\Omega(x_k + \alpha_k d) < +\infty\}$. Call L_{k+1} `= add(`L_k`,`P_k`)` to eventually 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 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: $\alpha_{k,new} \in [\beta_1\alpha_k, \beta_2\alpha_k]$ and replace the poll point $(x_k; \alpha_k; r_k; 1)$ in L_{k+1} by $(x_k; \alpha_{k,new}; r_k; 1)$.

During the optimization process a list of feasible points will be kept. Each point is stored in this list, jointly with the corresponding step size parameter, α_k, the comparison radius, r_k, and its classification as active or inactive, corresponding to setting the index i_k equal to 1 or 0, respectively. A point, generated in any of the two steps of the algorithm, is added to the list as an active point when:

- 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;
- the new point, x_{new}, is comparable with at least one of the stored points, y, meaning $\|x_{new} - y\| \leq r_y$, it is better than at least one of the points to which it is comparable with, meaning $f(x_{new}) \leq f(y) - \bar{\rho}(\alpha_y)$, and no point to which it is comparable with equals or decreases the corresponding objective function value, $f(y) < f(x_{new}) - \bar{\rho}(\alpha_y)$.

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 [17]), when a sufficient decrease strategy is adopted. Typical examples of forcing functions are $\rho(t) = t^{1+a}$, for $a > 0$.

There is a third possibility for adding a new point to the list, in this case as inactive. This situation occurs when a new point is comparable with an already stored active point, presenting a better objective function value than it, meaning

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

but another point already stored, comparable with the new one, equals or decreases this value.

Several methods could be considered for defining the search step in GLODS: random sampling [26], Latin hypercube sampling [20], Sobol sequences or Halton sequences [16]. 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 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 for performing it at every iteration. Several strategies could be implemented regarding this decision and eventually 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 launched at each iteration where the number of active points in the list is less or equal than this value.

The poll step starts by ordering the active points in the list, and selecting one of them has 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 [8], scaled by the step size parameter, α_k. In this procedure, complete or opportunistic approaches can be taken.

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. Otherwise, if generated at the poll step, both parameters will be equal to the step size of the poll center. 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. A schematic description of these procedures can be found in Algorithm 2.2.

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 will remain active in the list. This procedure allows the merging of direct searches with different initializations, when sufficiently close to each other.

Differently from a classic direct search method of directional type, GLODS generates three different 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 inactive points are added to the list.

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 equal 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

Algorithm 2.2: $[L_3]$=add(L_1, L_2)

for all $(x;\alpha_x;r_x;0) \in L_2$

do
- **if** $\min_{y\in L_1}(\|x-y\| - r_y) > 0$
  - **then** $L_1 = L_1 \cup \{(x;\alpha_0;r_0;1)\}$
  - **else**
    - **if** $x \notin L_1$
      - **then**
        - set $\alpha_a = 0$, $r_a = 0$, $i_{dom} = 0$, $p_{dom} = 0$ and $i_{comp} = 0$
        - **for** all $(y;\alpha_y;r_y;i_y) \in L_1$
        - **do**
          - **if** $\|x-y\| - r_y \leq 0$
            - **then**
              - **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**
                      - $\alpha_a = \alpha_y$
                      - $r_a = r_y$
              - **else if** $f(y) \leq f(x) - \bar{\rho}(\alpha_y)$
                - **then** $p_{dom} = 1$
        - **if** $p_{dom} = 0$
          - **then** $i_x = 1$
        - **if** $(i_{dom} > 0$ **or** $(p_{dom} = 0$ **and** $i_{comp} = 1))$
          - **then**
            - **if** $\alpha_x = 0$
              - **then** $L_1 = L_1 \cup \{(x;\alpha_a;r_a;i_x)\}$
              - **else** $L_1 = L_1 \cup \{(x;\alpha_x;r_x;i_x)\}$

$L_3 = L_1$

FIG. 2.1. *Procedure for adding new points, stored in L_2, to the current list, L_1.*

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;

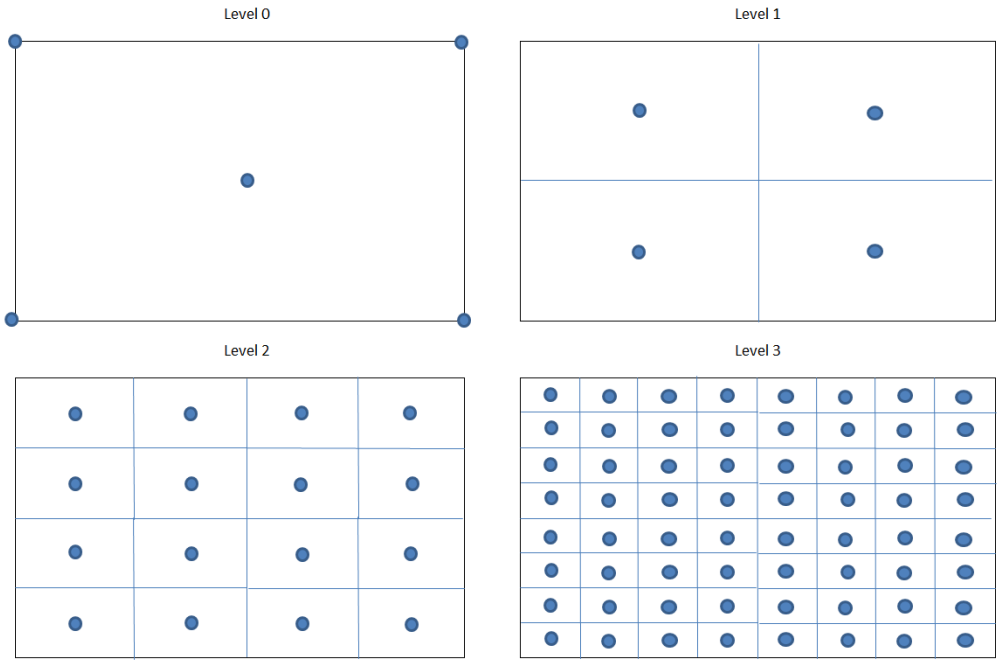

FIG. 2.2. *Four different levels of the strategy* 2^n-Centers *for a bound constrained problem of dimension* $n = 2$.

- avoid randomness: guaranteeing equal algorithmic performance for different runs, of 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 2.2 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, 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. To avoid this situation, each time the search step is performed, the strategy 2^n-*Centers* selects only 2^n of the box centers, not yet considered, one located in each of the new boxes 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 a subset of 2^n).

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 of the set* $\{z \in \mathbb{R}^n : z = arg\min_{w \in L} f(w)\}$ *is active.*

Proof. Suppose not. Let $z = arg\min_{w \in L} f(w)$ 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 later 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 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) [3] or Mesh Adaptive Direct Search (MADS) [4]) 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 [17]).

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 [3, 17].

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 directions $\mathcal{D}$, 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 [4].

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* $d_k \in D_k \subseteq \mathcal{D}$ *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 [4], 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 [28], in the context of pattern search, and generalized by Audet and Dennis to GPS [3] and MADS [4].

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. 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 (eventually 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 [17] 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 $y \in L_k$ 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 (eventually being 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 compact 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 $y \in L_k$, such that $\|x - y\| - r_y \leq 0$ and $f(x) < f(y) - \rho(\alpha_y) \leq f(y) - \rho(\alpha_*)$. The value of f would decrease by at least $\rho(\alpha_*) > 0$ at each successful iteration $k \geq k^*$. 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 [3] 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 [7]).

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 [4]).

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 [13], computed for a limit point of the sequence of iterates generated by the algorithm, for the whole directions belonging to the Clarke generalized tangent cone to the feasible region [6].

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 [6], $T_\Omega^{Cl}(x_*)$, and of Clarke-Jahn generalized directional derivatives [13], $f^\circ(x_*; d)$.

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., [23, 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*

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

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

Assuming the Lipschitz continuity of f near x, the Clarke-Jahn generalized deriva-

tives 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 [4] 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_*, then the previous definition of Clarke stationarity (Definition 3.7) can be restated using the gradient vectors.

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 a refining direction d for x_* in $T_\Omega^H(x_*)$. 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:

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

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 r_k is bounded above by $\nu||d - d_k/\|d_k\|\|$, where ν is the Lipschitz constant of f near x_*. Note, also, 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 a refining direction d for x_* in $T_\Omega^H(x_*)$. Assume that f is strictly differentiable at x_*. Then $\nabla f(x_*)^\top d \geq 0$.*

By assuming the 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 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 [4])

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

The second statement of the theorem results trivially. □

4. Numerical experiments. A basic version of the proposed algorithm was implemented in MATLAB, and numerically tested in a set comprising 63 bound constrained minimization problems collected from the 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 considered, 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 present a symmetric feasible region. Typically the global minimum for each problem will be located in the center of it. Since the inclusion of points near 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 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 problems variables. Additionally, it also considers the center of the feasible region. The algorithm performs opportunistic polling, considering the

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 [24, 12]	2	$[-5\ 0]^\top$	$[10\ 15]^\top$	3	3
cauchy [5]	4	3	17	–	–
cauchy [5]	10	2	26	–	–
cosine mixture [5]	2	-1	1	–	–
cosine mixture [5]	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 [5]	2	-1	1	–	1
exponencial [5]	4	-1	1	–	1
fifteenn local minima [5]	2	-10	10	15^2	1
fifteenn local minima [5]	4	-10	10	15^4	1
fifteenn local minima [5]	6	-10	10	15^6	1
fifteenn local minima [5]	8	-10	10	15^8	1
fifteenn local minima [5]	10	-10	10	15^{10}	1
fletcher powell [5]	3	-10	10	–	1
goldstein price [5]	2	-2	2	4	1
griewank [27, 11]	5	-600	600	–	1
griewank [27]	10	-400	400	–	1
gulf [1]	3	$[0.1\ 0\ 0]^\top$	$[100\ 25.6\ 5]^\top$	–	1
hartman 4 [5]	3	0	1	4	1
hartman 4 [5]	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
meyer roth [5]	3	-10	10	–	1
miele cantrell [5]	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 the number of known local minimizers and glob the number of known global minimizers.

positive basis $[I - I]$, 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 lowest 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}.

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 [5]	2	$[1\ 1]^\top$	$[21\ 8]^\top$	–	–
powell [1]	4	-10	10	1	1
rastrigin [1]	10	-5.12	5.12	–	1
rosenbrock [5, 12]	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 [5]	4	0	10	5	1
shekel 47 [5]	4	0	10	7	1
shekel 410 [5]	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 [5, 12]	2	$[-3\ -2]^\top$	$[3\ 2]^\top$	6	2
sphere [27, 11]	3	-5.12	5.12	1	1
storn tchebychev [1]	9	-128	128	–	1
tenn local minima [5]	2	-10	10	10^2	1
tenn local minima [5]	4	-10	10	10^4	1
tenn local minima [5]	6	-10	10	10^6	1
tenn local minima [5]	8	-10	10	10^8	1
three-hump-camel [1]	2	-5	5	3	1
transistor [1]	9	-10	10	–	1
wood [5]	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 the number of known local minimizers and glob the number of known global minimizers.

As comparison tool we considered the data profiles, proposed by Moré and Wild [21] for accessing the performance of derivative-free optimization solvers when there are constraints on the computational budget. 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

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

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 [26];
- Latin hypercube sampling [20];
- Sobol sequences [16];
- Halton sequences [16];
- 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 eventually 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, eventually initializing new direct searches in regions not yet explored. Other criteria could have been implemented, again traducing 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, 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.

From the analysis of the profiles, we can say that considering 2^n-Centers or Sobol sequences to define the search step in GLODS present 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 could not be enough to avoid the proximity of the initialization from the global minimizers.

To better access the impact of the implementation of different strategies in the search step of the algorithm, the experiments were repeated, considering the initialization with n random points. The corresponding data profiles are reported in Figures 4.3 and 4.4.

The advantage of defining a search step is now clearer. Strategy Halton positively distinguishes from the remaining ones. 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:

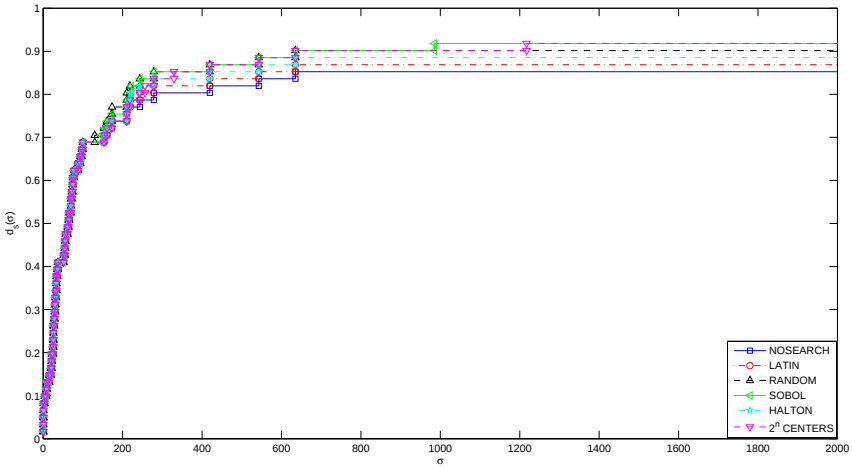

FIG. 4.1. *Comparing different strategies for defining the search step in GLODS (worst runs).*

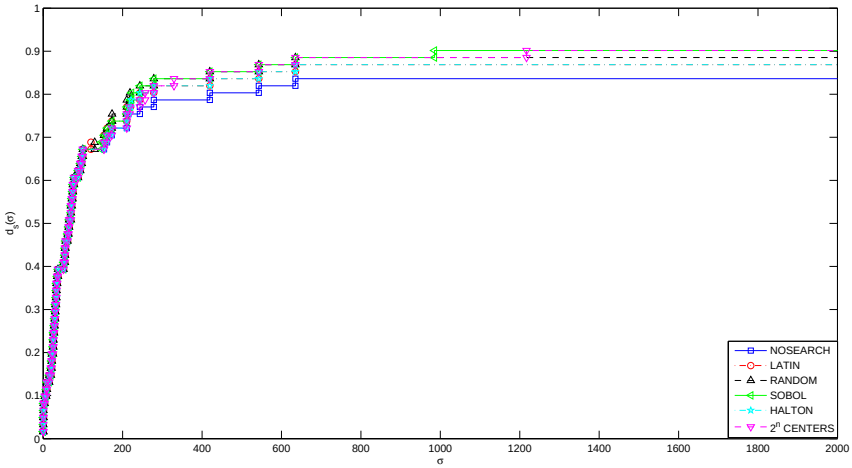

FIG. 4.2. *Comparing different strategies for defining the search step in GLODS (best runs).*

1. DIRECT – DIviding RECTangles [9];
2. MCS – Global optimization by Multilevel Coordinate Search [11].

Solvers were run with default values, for a maximum of 20000 function evaluations. As mentioned before, GLODS would additionally stop if the maximum value for the step size parameter corresponding to active points is smaller than 10^{-8}.

In this case, reporting a best and worst case is not necessary since all the solvers are deterministic. Results can be found in Figure 4.5.

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

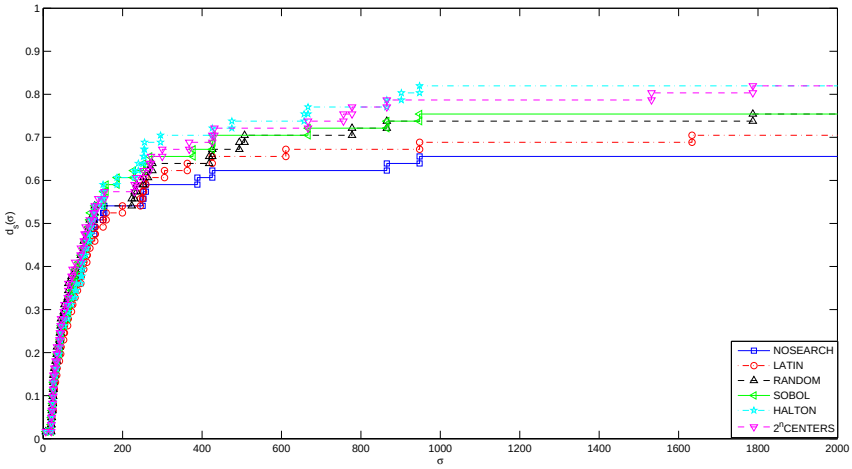

FIG. 4.3. *Comparing different strategies for defining the search step in GLODS, with random initialization (worst runs).*

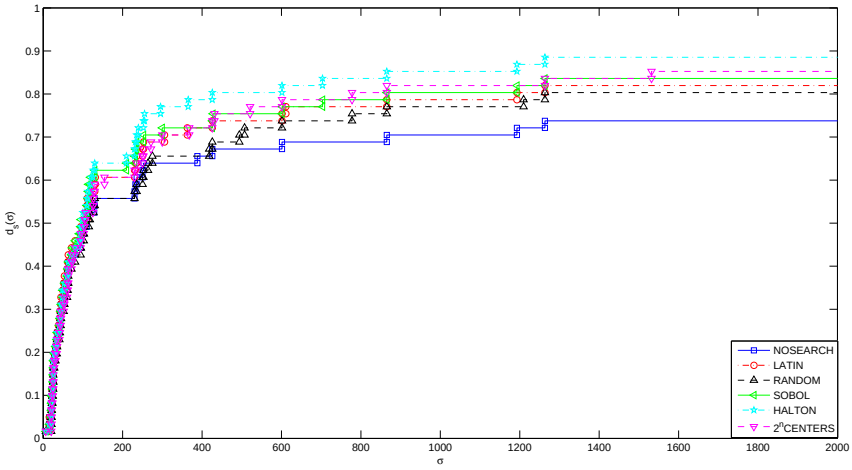

FIG. 4.4. *Comparing different strategies for defining the search step in GLODS, with random initialization (best runs).*

competitive, presenting the best performance for budgets of moderate size.

5. Conclusions. This work can be regarded as a first attempt of generalizing direct search to global optimization. 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

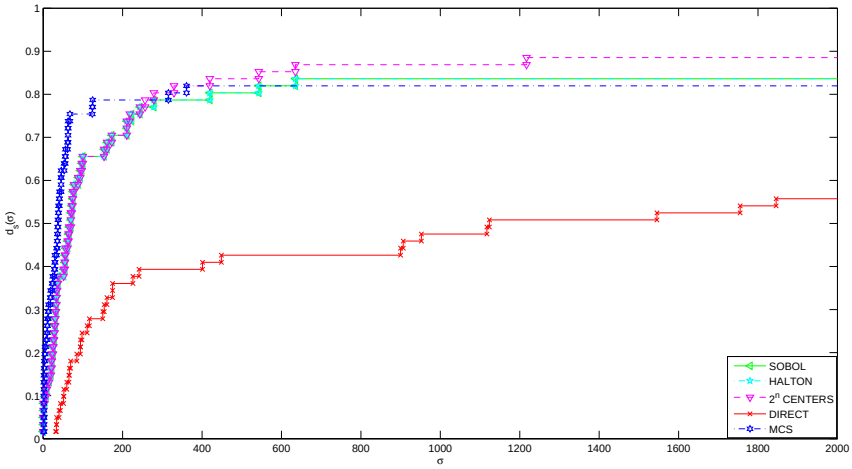

FIG. 4.5. *Comparing GLODS with other solvers.*

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 [29].

A basic numerical implementation of GLODS algorithm has shown to be competitive with well-established global 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 the 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 and J. E. Dennis Jr. Analysis of generalized pattern searches. *SIAM J. Optim.*, 13:889–903, 2003.

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

[5] 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.

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

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

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

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

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

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

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

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

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

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

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

[17] 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.

[18] 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.

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

[20] 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.

[21] 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.

[22] 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.

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

[24] 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.

[25] 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.

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

[27] 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.

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

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