# SMGO: A Set Membership Approach to Data-Driven Global Optimization

Lorenzo Sabug Jr.*, Fredy Ruiz, Lorenzo Fagiano**

*Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano*
*Piazza Leonardo da Vinci 32, 20133 Milano, Italy*



**Abstract**

Many science and engineering applications feature non-convex optimization problems where the objective function can not be handled analytically, i.e. it is a black box. Examples include design optimization via experiments, or via costly finite elements simulations. To solve these problems, global optimization routines are used. These iterative techniques must trade-off exploitation close to the current best point with exploration of unseen regions of the search space. In this respect, a new global optimization strategy based on a Set Membership (SM) framework is proposed. Assuming Lipschitz continuity of the cost function, the approach employs SM concepts to decide whether to switch from an exploitation mode to an exploration one, and vice-versa. The resulting algorithm, named SMGO (Set Membership Global Optimization) is presented. Theoretical properties regarding convergence and computational complexity are derived, and implementation aspects are discussed. Finally, the SMGO performance is evaluated on a set of benchmark non-convex problems and compared with those of other global optimization approaches.

## 1. Introduction

In many science and engineering fields, such as mechanical design, fluid-dynamics, electromagnetics, multi-physics simulations, control systems tuning, and chemical experiments, black-box optimization problems arise. Black-box functions are named as such due to the fact that an explicit mathematical model is unavailable, or too complicated to be handled analytically. In these cases, the optimization strategy can rely only on function values obtained through empirical tests. Moreover, these problems may present several local minima, and the time and resources required to carry out a single function evaluation are rather large, so that the solution method shall make the most efficient use of the available trials. To highlight this fact, we refer to the process of performing a new test as *long function* evaluation.

Seeking a minimizer in this framework is referred to as black-box optimization [1], or derivative-free optimization [2]. This is in contrast for example to gradient-based and Newton-type methods where the (first, or even further) derivatives are also assumed to be available. If the black-box function is assumed differentiable, one could resort to gradient estimation techniques to still employ these methods, which however require a rather large number of long function evaluations to estimate a local quantity (i.e., the gradient at the currently evaluated point) [3] and are designed to converge to a local optimum by always improving from the initial guess, not to explore the decision space searching for a global solution.

Due to time/resource limitations, in global optimization there are two conflicting aspects that must be considered when choosing the next test point for a long function evaluation [3]. The first, *exploitation*, pertains to improving the result by testing more points in the vicinity of the current best known one. The second, *exploration*, aims to gain

*Corresponding author. The first author would like to acknowledge the support of the Department of Science and Technology–Science Education Institute (DOST-SEI) of the Philippines for his research.

**This research has been supported by the Italian Ministry of University and Research (MIUR) under the PRIN 2017 grant n. 201732RS94 "Systems of Tethered Multicopters".

*Email addresses:* lorenzojr.sabug@polimi.it (Lorenzo Sabug Jr.), fredy.ruiz@polimi.it (Fredy Ruiz), lorenzo.fagiano@polimi.it (Lorenzo Fagiano)

more information about the cost function in other regions of the search space, seeking other, possibly better, minima. Exploitation and exploration strategies are at the core of most global optimization approaches, including stochastic global search, response surface, and Lipschitz-based methods.

Stochastic global search algorithms, such as random search methods [4], particle swarm algorithms [5], and its offshoot variants [6, 7], use a population of search points per iteration. According to the corresponding function values, heuristics are applied to calculate the search point locations for the next iteration (or "generation"). These methods are widespread but require numerous long function evaluations per generation, which can be impractical. Moreover, aspects such as guaranteed convergence and optimality gap may be difficult to analyze. Response surface methods, on the other hand, generate an approximation of the black-box function via, e.g., Gaussian process regression [1], radial basis functions [8, 9], or neural networks [10]. These approximations can then be used for the exploitation or exploration routines, whichever is appropriate. Albeit the number of long function evaluation is lower in this case, optimality gap is again not easily assessed, and there is an additional exponential computational burden to derive and refine the approximated objective function.

Another category of black-box optimizers are so-called Lipschitz-based algorithms. They rely on the assumption that the long function is Lipschitz continuous. An early method, proposed independently by Piyavskii [11] and Shubert [12], uses a known Lipschitz constant to draw the lower bounds of the black-box function given the existing samples, starting with the bounds (corners) of the search space. In this context, the Piyavskii-Shubert ("sawtooth") method chooses the next point by searching the location of the minimum lower bound. However, the method is proposed for one-dimensional function optimization only; furthermore, Lipschitz constants are rarely known in practical cases, and need to be estimated in the process.

A later proposal named DIRECT [13, 14, 15] is a modification to Piyavskii-Shubert method, removing the need for a known Lipschitz constant and starting samples from the corners. Instead, it invariably performs a first sample at the center of the search space, which is afterwards subdivided into three intervals. The Lipschitz constant estimate is used to draw the lower bounds, which are used to select potentially optimal intervals (hyperrectangles). Such hyperrectangles are iteratively subdivided, and their corresponding centers are sampled by batch, and potential intervals for sampling are selected again. This cycle proceeds until the maximum number of iterations is reached, with each iteration performing multiple long function evaluations. An offshoot method named DISIMPL [16, 17] employs simplices instead of hyperrectangles, thus extending the usage from rectangular search spaces only to more general polytopic ones. On one hand, DIRECT and DISIMPL are completely deterministic and reproducible (assuming no noise on function evaluation), however, they only admit a fixed starting point, and do not propose how to handle existing sampled points at the beginning of the algorithms. Furthermore, the batch-based sampling means that the next set of trial points is not chosen until the current batch of sampling is finished, which can mean overlooking possible precious information added by individual samples as they come in.

LIPO and AdaLIPO [18] use a random binary variable to decide between exploitation and exploration; the only difference being that the former assumes a fixed Lipschitz constant, while the latter estimates it from existing data. For exploitation, they choose a random point within the set of potential optimizing points, i.e. those whose lower bound on the black-box function is below the current best sample. For exploration, it chooses a random point from the entire search space. These algorithms, owing to the simple implementation, are fast in terms of choosing the next sampling point, and can easily scale to high dimensional problems without significant hit on computational burden. However, such a completely randomized exploitation/exploration sampling may be inefficient in finding new local minima, because it does not fully utilize the shape of lower bounds. Furthermore, the results of such an algorithm are unrepeatable because of its randomized design.

In the described context, we propose a new Lipschitz-based algorithm that exploits Set Membership (SM) nonlinear function approximation theory [19]. So far, SM theory proved effective for nonlinear system identification [20], filter design [21], and controller design [22], among others. Here, we employ SM theory for the first time in global optimization. Our paper delivers the following contributions:

- The Set Membership Global Optimization

(SMGO) algorithm for Lipschitz-continuous black-box functions is introduced. It employs the concepts of SM lower- and upper bounds in exploitation and exploration routines. Unlike the above-mentioned Lipschitz-based algorithms, we address the exploration problem systematically, by casting it as minimization of the uncertainty as inferred using SM techniques. Furthermore, our proposed algorithm is completely repeatable, and allows one to consider custom starting points and initial samples when available;

- Theoretical properties of SMGO are derived, including convergence, optimality gap, and computational complexity;

- Practical aspects are discussed, and ways to improve the computational efficiency are described. In particular, an iterative implementation is described, which exploits results from previous iterations to alleviate the computations at the current one;

- The SMGO algorithm is compared with other representative optimizers with different test functions, showing that the performance of SMGO is very competitive w.r.t. the state of the art, especially on functions with many global minima.

A preliminary version of this work appeared in [23]. With respect to that contribution, the algorithm presented here is more efficient, moreover the theoretical results and implementation analysis are new, and more benchmark results are presented. The MATLAB code for SMGO, as well as the other scripts used to generate the results, are available on GitHub under the following URL: `https://github.com/lorenzosabugjr/smgo`.

This paper is organized in seven sections. Section 2 gives the general problem statement and assumptions. Section 3 describes the proposed SMGO algorithm. The convergence properties and optimality gap calculations are discussed in Section 4, with implementation notes in Section 5. Section 6 compares the performance of the proposed algorithm with other global optimization techniques on representative test functions, and Section 7 concludes this paper and provides insights on future directions.

## 2. Problem Statement

Consider a cost function $z = f_o(\boldsymbol{x}),\ f_o : \mathcal{X} \to \mathbb{R}$, where $x \in \mathcal{X}$ is the vector of decision variables, $\mathcal{X} \subset \mathbb{R}^D$ is a compact and convex polytope ("search set"). No analytical form of $f_o$ is assumed available. The *a priori* knowledge about $f_o$ is given by the following assumption:

**Assumption 1.** *$f_o$ is Lipschitz continuous with unknown Lipschitz constant $\gamma_o$, i.e.,*

$$f_o \in \mathcal{F}(\gamma_o)$$

*where*

$$\mathcal{F}(\gamma_o) \doteq \Big\{ f \in \boldsymbol{C}^0(\mathcal{X}) : |f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)| \leq \gamma_o \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|, \forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{X} \Big\}$$

Being a Lipschitz continuous function on a compact set, $f_o$ presents a global minimum $z^*$:

$$z^* = \min_{\boldsymbol{x} \in \mathcal{X}} f_o(\boldsymbol{x}). \tag{1}$$

Let us denote with

$$\mathcal{X}^* = \{\ \boldsymbol{x} \in \mathcal{X} \mid f_o(\boldsymbol{x}) = z^*\ \} \tag{2}$$

the corresponding set of global minimizers.

We further assume that it is possible to acquire information about $f_o$ by sampling (long function evaluation):

**Assumption 2.** *Given a vector of decision variables $\boldsymbol{x} \in \mathcal{X}$, it is possible to sample the cost function without noise:*

$$z = f_o(\boldsymbol{x}).$$

The set of collected data is denoted as:

$$\boldsymbol{X}^{\langle n \rangle} = \Big\{ (\boldsymbol{x}^{(1)}, z^{(1)}); (\boldsymbol{x}^{(2)}, z^{(2)}); \ldots; (\boldsymbol{x}^{(n)}, z^{(n)}) \Big\} \tag{3}$$

where $n \in \mathbb{N}$ is the number of data points and $z^{(i)} = f_o(\boldsymbol{x}^{(i)})$. For simplicity, with a slight abuse of notation, we will also write $\boldsymbol{x}^{(i)} \in \boldsymbol{X}^{\langle n \rangle}$ when the pair $(\boldsymbol{x}^{(i)}, z^{(i)})$ belongs to the data set (3).

We denote with $(\boldsymbol{x}^{*\langle n \rangle}, z^{*\langle n \rangle})$ the best pair in $\boldsymbol{X}^{\langle n \rangle}$, where

$$(\boldsymbol{x}^{*\langle n \rangle}, z^{*\langle n \rangle}) = \underset{(\boldsymbol{x}^{(k)}, z^{(k)}) \in \boldsymbol{X}^{\langle n \rangle}}{\arg\min} z^{(k)} \tag{4}$$

If the result of (4) is not unique, a lexicographic criterion is used to sort the minimizers, and the first one is picked. We further denote with $\delta^{\langle n \rangle}$ the optimality gap:

$$\delta^{\langle n \rangle} = z^{*\langle n \rangle} - z^*. \tag{5}$$

We assume that a starting number $n_0 \geq 1$ of data points is also available, for example collected by the user in a first testing campaign, forming the set $\boldsymbol{X}^{\langle n_0 \rangle} \subset \mathcal{X}$.

We can now state the problem addressed in this paper.

**Problem 1.** *Design an algorithm that, under Assumptions 1-2, generates a sequence of points* $\{\boldsymbol{x}^{(n_0+1)}, \boldsymbol{x}^{(n_0+2)}, \ldots\}$, $\boldsymbol{x}^{(i)} \in \mathcal{X}$, *such that:*

$$\forall \epsilon > 0,\ \exists n_\epsilon < \infty : z^{*\langle n_\epsilon \rangle} \leq z^* + \epsilon. \tag{6}$$

*Moreover, for a finite sequence length $N$, provide a method to compute a bound on the optimality gap* $\delta^{\langle N \rangle}$.

## 3. Set Membership Global Optimization (SMGO): Algorithm

SMGO addresses Problem 1 with a sequential procedure, as common in global optimization [1, 13, 18]. At each iteration $n \geq n_0$, the next test point $\boldsymbol{x}^{(n+1)}$ is chosen with a strategy that exploits the prior knowledge on function $f_o$, given by Assumption 1, and the current data set $\boldsymbol{X}^{\langle n \rangle}$. In particular, a valid (i.e., consistent with data) estimate of the Lipschitz constant $\gamma_o$ is derived at $n_0$ and updated at each $n > n_0$, in order to estimate upper and lower bounds on $f_o$. SMGO then leverages the latter to compute $\boldsymbol{x}^{(n+1)}$, choosing between an exploitation mode and an exploration one.

### *3.1. Preliminaries: Lipschitz constant estimation, cost function bounds, and search mesh*

At the start of each iteration ($n > n_0$), one long function evaluation $z^{(n)} = f_o(\boldsymbol{x}^{(n)})$ is carried out, where $\boldsymbol{x}^{(n)}$ is the test point selected at the previous iteration. This resulting new data point $(\boldsymbol{x}^{(n)},\ z^{(n)})$ is added to the set of collected samples:

$$\boldsymbol{X}^{\langle n \rangle} = \boldsymbol{X}^{\langle n-1 \rangle} \cup (\boldsymbol{x}^{(n)},\ z^{(n)}).$$

At the first iteration, $n = n_0$, an initial estimate of the Lipschitz constant, $\gamma^{\langle n_0 \rangle}$, is computed as:

$$\gamma^{\langle n_0 \rangle} = \max_{i,j=1,\ldots,n_0,\, i \neq j} \frac{|z^{(i)} - z^{(j)}|}{\|\boldsymbol{x}^{(i)} - \boldsymbol{x}^{(j)}\|} \tag{7}$$

If $n_0 = 1$, the Lipschitz constant estimate can be initialized as an arbitrarily small number. When $n > n_0$, the minimum feasible Lipschitz constant estimate $\gamma^{\langle n \rangle}$ is updated as follows:

$$\gamma^{\langle n \rangle} = \max\left(\gamma^{\langle n-1 \rangle}, \max_{k=1,\ldots,n-1} \frac{|z^{(n)} - z^{(k)}|}{\|\boldsymbol{x}^{(n)} - \boldsymbol{x}^{(k)}\|}\right). \tag{8}$$

Such an estimate is guaranteed to be compatible with the available data and prior information on $f_o$, given Assumption 2 [20]. Moreover, $\gamma^{\langle n \rangle}$ is monotonically increasing with $n$ and convergence to the true Lipschitz constant $\gamma_o$ is guaranteed if the test points densely cover the search set [22]. This is the case for the SMGO algorithm as shown in Section 4.

Given the data set $\boldsymbol{X}^{\langle n \rangle}$ and Lipschitz constant estimate $\gamma^{\langle n \rangle}$ we can compute the following lower and upper bounds on $f_o$ [20]:

$$\underline{z}^{\langle n \rangle}(\boldsymbol{x}) \triangleq \max_{k=1,\ldots,n} \left(z^{(k)} - \mu\gamma^{\langle n \rangle}\|\boldsymbol{x} - \boldsymbol{x}^{(k)}\|\right), \tag{9}$$

$$\overline{z}^{\langle n \rangle}(\boldsymbol{x}) \triangleq \min_{k=1\ldots n} \left(z^{(k)} + \mu\gamma^{\langle n \rangle}\|\boldsymbol{x} - \boldsymbol{x}^{(k)}\|\right). \tag{10}$$

When $\mu = 1$, the functions $\underline{z}^{\langle n \rangle}$, $\overline{z}^{\langle n \rangle}$ represent, respectively, the minimum and maximum values that the objective function can take for a given vector of decision variables $\mathbf{x} \in \mathcal{X}$, given the information collected up to iteration $n$.

In SMGO, an (user-defined) overestimation parameter $\mu > 1$ is used in (9)-(10) to compensate for the fact that the $\gamma^{\langle n \rangle}$ estimated from the currently available data points is an underestimator of the true Lipschitz constant $\gamma_o$.

The set

$$\mathbb{F}^{\langle n \rangle} = \left\{f \in \mathcal{F}(\gamma^{\langle n \rangle}) : \underline{z}^{\langle n \rangle}(\boldsymbol{x}) \leq f(\boldsymbol{x}) \leq \overline{z}^{\langle n \rangle}(\boldsymbol{x})\right\}$$

is referred to as the *feasible functions set* [20] at iteration $n$. The functions $\underline{z}^{\langle n \rangle}(\boldsymbol{x})$ and $\overline{z}^{\langle n \rangle}(\boldsymbol{x})$ are intersections of hypercones (referred simply as cones

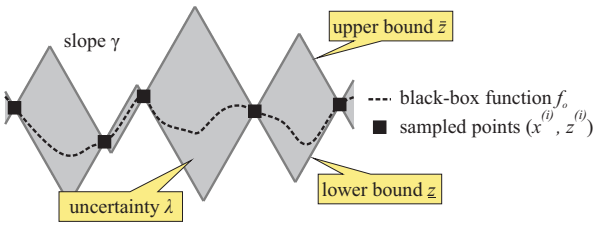


Figure 1: Scalar example of optimal lower and upper bounds $\underline{z}^{\langle n \rangle}(\boldsymbol{x})$, $\overline{z}^{\langle n \rangle}(\boldsymbol{x})$ from finite samples of $f_o$, and of the corresponding uncertainty interval $\lambda^{\langle n \rangle}(\boldsymbol{x})$.

in the remainder), each one with vertex at a tested point. A representation of these bounds is depicted in Fig. 1 for a function $f_o$ with scalar input argument. It should be emphasized that $\underline{z}^{\langle n \rangle}$ and $\overline{z}^{\langle n \rangle}$ are calculated from the data in $\boldsymbol{X}^{\langle n \rangle}$, without any need for additional evaluations of $f_o$.

The uncertainty about the value of $f_o$ at the point $\boldsymbol{x} \in \mathcal{X}$ is computed as:

$$\lambda^{\langle n \rangle}(\boldsymbol{x}) = \overline{z}^{\langle n \rangle}(\boldsymbol{x}) - \underline{z}^{\langle n \rangle}(\boldsymbol{x}). \tag{11}$$

The SMGO algorithm proceeds in two steps. First, an exploitation routine searches for a local minimizer of $\underline{z}^{\langle n \rangle}$. Then, an exploration step seeks a maximizer of $\lambda^{\langle n \rangle}$. Exact solutions for such subproblems can be computed by utilizing Hyperbolic Voronoi Diagrams (HVDs) [20, 24], however at rather high computational cost. To increase computational efficiency, we thus introduce a mesh given by segments $l_{i,j}$ connecting any two points $\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)} \in \boldsymbol{X}^{\langle n \rangle}$:

$$l_{i,j} = \Big\{ \boldsymbol{x} = a\boldsymbol{x}^{(i)} + (1-a)\boldsymbol{x}^{(j)} : a \in [0,1], \\ \boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)} \in \boldsymbol{X}^{\langle n \rangle},\, i \neq j \Big\} \tag{12}$$

Such a mesh allows us to derive analytic solutions to the computation of the next test point $\boldsymbol{x}^{(n+1)}$, according to the two strategies presented next.

### *3.2. Exploitation (Mode θ)*

Consider the best data point $(\boldsymbol{x}^{*\langle n \rangle}, z^{*\langle n \rangle})$ (4) at iteration $n$. Mode $\theta$ searches a point $\boldsymbol{x}_{\theta*}^{\langle n \rangle}$ with the highest predicted improvement w.r.t. $\boldsymbol{x}^{*\langle n \rangle}$, according to the lower bound $\underline{z}^{\langle n \rangle}$ (9). Specifically, the exploitation routine considers the set of segments $l_{n^*,i}$ between $\boldsymbol{x}^{*\langle n \rangle}$ and any other sampled point $\boldsymbol{x}^{(i)} \in \boldsymbol{X}^{\langle n \rangle}$ to search for the candidate point $\boldsymbol{x}_{\theta*}^{\langle n \rangle}$. Here, index $n^*$ is the data-point index of the pair $(\boldsymbol{x}^{*\langle n \rangle}, z^{*\langle n \rangle})$ in the data set $\boldsymbol{X}^{\langle n \rangle}$.

We first evaluate the lower bounds on $f_o(\boldsymbol{x}) : \boldsymbol{x} \in l_{n^*,i}$ considering only the hypercones defined by the data points $(\boldsymbol{x}^{*\langle n \rangle}, z^{*\langle n \rangle})$ and $(\boldsymbol{x}^{(i)}, z^{(i)})$, i.e.:

$$z^{*\langle n \rangle} - \mu\gamma^{\langle n \rangle}\|\boldsymbol{x} - \boldsymbol{x}^{*\langle n \rangle}\| \tag{13}$$

and

$$z^{(i)} - \mu\gamma^{\langle n \rangle}\|\boldsymbol{x} - \boldsymbol{x}^{(i)}\|. \tag{14}$$

The intersection of these two cones on the segment $l_{n^*,i}$ is the smallest feasible value of $f_o(\boldsymbol{x}) : \boldsymbol{x} \in l_{n^*,i}$ considering the information given by the two points $(\boldsymbol{x}^{*\langle n \rangle}, z^{*\langle n \rangle})$, $(\boldsymbol{x}^{(i)}, z^{(i)})$ and the Lipschitz constant estimate $\gamma^{\langle n \rangle}$. Such an intersection, denoted with $\boldsymbol{x}_\theta^{(i)}$, can be derived analytically on the basis of geometrical considerations (see Fig. 2). In fact, denoting

$$s_i = \frac{z^{(i)} - z^{*\langle n \rangle}}{\|\boldsymbol{x}^{(i)} - \boldsymbol{x}^{*\langle n \rangle}\|} \geq 0, \tag{15}$$

it follows that

$$\boldsymbol{x}_\theta^{(i)} = \boldsymbol{x}^{*\langle n \rangle} + \frac{1 - \frac{s_i}{\mu\gamma^{\langle n \rangle}}}{2}(\boldsymbol{x}^{(i)} - \boldsymbol{x}^{*\langle n \rangle}). \tag{16}$$

By evaluating (16) for all possible segments, we obtain the set $\boldsymbol{X}_\theta^{\langle n \rangle} = \{\boldsymbol{x}_\theta^{(i)}, i = 1, \ldots, n,\, i \neq n^*\}$. Then $\boldsymbol{x}_{\theta*}^{\langle n \rangle}$ is chosen as:

$$\boldsymbol{x}_{\theta*}^{\langle n \rangle} = \arg\min_{\boldsymbol{x} \in \boldsymbol{X}_\theta^{\langle n \rangle}} \underline{z}^{\langle n \rangle}(\boldsymbol{x}) \tag{17a}$$

s.t.

$$\underline{z}^{\langle n \rangle}(\boldsymbol{x}) = z^{*\langle n \rangle} - \mu\gamma^{\langle n \rangle}\|\boldsymbol{x} - \boldsymbol{x}^{*\langle n \rangle}\| \tag{17b}$$

where the constraint (17b) ensures that the point selected is such that the lower bound $\underline{z}^{\langle n \rangle}$ considering the information provided by all the sampled points coincides with the lower bound provided by the current best point $(\boldsymbol{x}^{*\langle n \rangle}, z^{*\langle n \rangle})$ alone. This usually happens in the vicinity of $\boldsymbol{x}^{*\langle n \rangle}$, in particular at the border of the hyperbolic Voronoi cell pertaining to $\boldsymbol{x}^{*\langle n \rangle}$, see [20]. The feasibility of problem (17) is always guaranteed, as shown in Lemma 1 in Section 4.

To decide whether $\boldsymbol{x}_{\theta*}^{\langle n \rangle}$ shall be eventually taken as the next test point $\boldsymbol{x}^{\langle n+1 \rangle}$ for the long function, or the exploration routine shall be called instead, the following condition is evaluated:

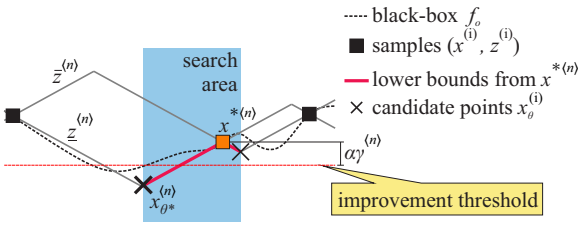


Figure 2: Qualitative example of the exploitation routine for a scalar problem.

$$\underline{z}^{\langle n\rangle}(\boldsymbol{x}_{\theta*}^{\langle n\rangle}) \leq z^{*\langle n\rangle} - \alpha\gamma^{\langle n\rangle}, \tag{18}$$

where $\alpha\gamma^{\langle n\rangle}$ is referred to as the *expected improvement threshold* and $\alpha \in [0, 1)$ is the SMGO *exploitation parameter*, adjusted by the user. If (18) is fulfilled (as in the example of Fig. 2), it means that the potential improvement provided by $\boldsymbol{x}_{\theta*}^{\langle n\rangle}$ is larger than the threshold. In this case, we set

$$\boldsymbol{x}^{(n+1)} = \boldsymbol{x}_{\theta*}^{\langle n\rangle}.$$

Otherwise, SMGO switches to exploration mode, described in the following subsection.

### *3.3. Exploration (Mode $\psi$)*

In this mode, the candidate point $\boldsymbol{x}_{\psi*}^{\langle n\rangle} \in \mathcal{X}$ is chosen as the one with largest $\lambda^{\langle n\rangle}$ value among selected candidates. Sampling $\boldsymbol{x}_{\psi*}^{\langle n\rangle}$ will yield $\lambda^{\langle n+1\rangle}(\boldsymbol{x}_{\psi*}^{\langle n\rangle}) = 0$ (due to noiseless function evaluation, see Assumption 2), and the uncertainty in its vicinity will correspondingly decrease.

Similar to Mode $\theta$, the search for $\boldsymbol{x}_{\psi*}^{\langle n\rangle}$ considers the segments among the samples in $\boldsymbol{X}^{\langle n\rangle}$, in order to limit the computational complexity. In this case, we consider as candidates all the midpoints between any pair $(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}) \in \boldsymbol{X}^{\langle n\rangle}$. The reason is that midpoints generally feature larger uncertainty than other points on the segments, which are closer to one of the two extremes. Hence, we form the set of midpoints:

$$\boldsymbol{X}_{\psi}^{\langle n\rangle} = \left\{ \boldsymbol{x}_{\psi}^{(i,j)} = \frac{\boldsymbol{x}^{(i)} + \boldsymbol{x}^{(j)}}{2} \;\middle|\; \boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)} \in \boldsymbol{X}^{\langle n\rangle} \right\}, \tag{19}$$

and, according to the described rationale, the exploration routine picks the next test point as:

$$\boldsymbol{x}^{(n+1)} = \boldsymbol{x}_{\psi*}^{\langle n\rangle} = \underset{\boldsymbol{x}\in\boldsymbol{X}_{\psi}^{\langle n\rangle}}{\arg\max}\, \lambda^{\langle n\rangle}(\boldsymbol{x}). \tag{20}$$

Also in this case, if (20) does not have a unique solution, a lexicographic criterion is added to select the maximizer.

### *3.4. Algorithm summary*

A summarized flow of the SMGO method is given as pseudo-code in Algorithm 1.

**Algorithm 1:** SMGO Algorithm

**Input:** Long function $f_o$, initial data points $\boldsymbol{X}^{\langle n_0\rangle}$, maximum number of long function evaluations $N$, exploitation parameter $\alpha$, overestimation parameter $\mu$

**while** $n_0 < n \leq N$ **do**
  // Long function evaluation, data update
  Evaluate the long function $f_o$ at $\boldsymbol{x}^{(n)}$ and measure output $z^{(n)}$. Update the data set $\boldsymbol{X}^{\langle n\rangle} \leftarrow \boldsymbol{X}^{\langle n-1\rangle} \cup (\boldsymbol{x}^{(n)}, z^{(n)})$
  Update current best sample $(\boldsymbol{x}^{*\langle n\rangle}, z^{*\langle n\rangle})$ and Lipschitz constant $\gamma^{\langle n\rangle}$ from $\boldsymbol{X}^{\langle n\rangle}$
  // Exploitation (Mode $\theta$)
  Evaluate lower bounds on segments around the current best sample $\boldsymbol{x}^{*\langle n\rangle}$ to choose the exploitation point $\boldsymbol{x}_{\theta*}^{\langle n\rangle}$
  **if** *expected improvement condition* (18) *is met* **then**
    Assign test point for next iteration $\boldsymbol{x}^{(n+1)} \leftarrow \boldsymbol{x}_{\theta*}^{\langle n\rangle}$
  **else**
    // Exploration (Mode $\psi$)
    Find $\boldsymbol{x}_{\psi*}^{\langle n\rangle}$ as the midpoint, over all possible segments, with largest uncertainty $\lambda^{\langle n\rangle}$
    Assign test point for next iteration $\boldsymbol{x}^{(n+1)} \leftarrow \boldsymbol{x}_{\psi*}^{\langle n\rangle}$
  Go to next iteration $n \leftarrow n + 1$
Final optimal point and value: get the best sample $(\boldsymbol{x}^{*\langle N\rangle}, z^{*\langle N\rangle})$ from the set $\boldsymbol{X}^{\langle N\rangle}$

### *3.5. On the choice of $\alpha$ and $\mu$*

The SMGO parameter $\alpha$ affects only Mode $\theta$: a smaller $\alpha$ value leads to higher tendency for exploitation. Regarding $\mu$, using a value close to one is recommended, because a higher Lipschitz constant overestimator leads to $\boldsymbol{x}_{\theta}^{(i)} \to \frac{\boldsymbol{x}^{*\langle n\rangle} + \boldsymbol{x}^{(i)}}{2}$, which would not fully utilize the Set Membership

approach in exploitation. At the same time, excessively large values of $\mu$ increase the conservativeness of $\lambda^{\langle n \rangle}$. Moreover, in the computation of $\lambda^{\langle n \rangle}(\boldsymbol{x}_{\psi}^{(i,j)})$, $\boldsymbol{x}_{\psi}^{(i,j)} \in \boldsymbol{X}_{\psi}^{\langle n \rangle}$ (19)-(20), with larger $\mu$ values the terms $\|\boldsymbol{x}_{\psi}^{(i,j)} - \boldsymbol{x}^{(i)}\|$, $i, j = 1, \ldots, n$ gain a higher relative importance with respect to the corresponding values of $z^{(i)}$ (see (9)-(10)), so that the exploration mode will tend to select the midpoints that are farthest away from the available data-points.

## 4. Theoretical Properties

We now analyze the convergence and optimality properties of the SMGO algorithm. First, we prove that problem (17) is always feasible, thus showing that the exploitation routine always returns a valid candidate next point $\boldsymbol{x}_{\theta *}^{\langle n \rangle}$.

**Lemma 1.** *At any iteration $n$, problem* (17) *admits at least one feasible point.*

*Proof.* Take the pair $(\boldsymbol{x}^{(i)}, z^{(i)})$ such that

$$z^{(i)} - \mu\gamma^{\langle n \rangle}\|\boldsymbol{x}^{*\langle n \rangle} - \boldsymbol{x}^{(i)}\| = \max_{(\boldsymbol{x},z)\in\boldsymbol{X}^{\langle n \rangle}\setminus(\boldsymbol{x}^{*\langle n \rangle}, z^{*\langle n \rangle})} z - \mu\gamma^{\langle n \rangle}\|\boldsymbol{x}^{*\langle n \rangle} - \boldsymbol{x}\|, \tag{21}$$

and consider the corresponding candidate exploitation point $\boldsymbol{x}_{\theta}^{(i)}$ given by (16). By construction we have that

$$z^{(i)} - \mu\gamma^{\langle n \rangle}\|\boldsymbol{x}_{\theta}^{(i)} - \boldsymbol{x}^{(i)}\| = z^{*\langle n \rangle} - \mu\gamma^{\langle n \rangle}\|\boldsymbol{x}_{\theta}^{(i)} - \boldsymbol{x}^{*\langle n \rangle}\|,$$

and we want now to prove that this point also coincides with $\underline{z}^{\langle n \rangle}(\boldsymbol{x}_{\theta}^{(i)})$, i.e. that:

$$z^{(i)} - \mu\gamma^{\langle n \rangle}\|\boldsymbol{x}_{\theta}^{(i)} - \boldsymbol{x}^{(i)}\| = \max_{(\boldsymbol{x},z)\in\boldsymbol{X}^{\langle n \rangle}} z - \mu\gamma^{\langle n \rangle}\|\boldsymbol{x}_{\theta}^{(i)} - \boldsymbol{x}\|. \tag{22}$$

Assume now, for the purpose of contradiction, that:

$$\exists(\boldsymbol{x}^{(j)}, z^{(j)}) \neq (\boldsymbol{x}^{(i)}, z^{(i)}) : z^{(j)} - \mu\gamma^{\langle n \rangle}\|\boldsymbol{x}_{\theta}^{(i)} - \boldsymbol{x}^{(j)}\| > z^{(i)} - \mu\gamma^{\langle n \rangle}\|\boldsymbol{x}_{\theta}^{(i)} - \boldsymbol{x}^{(i)}\|, \tag{23}$$

thus invalidating (22). Then, denoting with $a = \frac{1}{2}(1 - \frac{s_i}{\mu\gamma^{\langle n \rangle}})$ (see (16)), we would have:

$$\begin{aligned} z^{(j)} - \mu\gamma^{\langle n \rangle}\|\boldsymbol{x}^{*\langle n \rangle} - \boldsymbol{x}^{(j)}\| \geq \\ z^{(j)} - \mu\gamma^{\langle n \rangle}\|\boldsymbol{x}^{*\langle n \rangle} - \boldsymbol{x}_{\theta}^{(i)}\| - \mu\gamma^{\langle n \rangle}\|\boldsymbol{x}_{\theta}^{(i)} - \boldsymbol{x}^{(j)}\| > \\ z^{(i)} - \mu\gamma^{\langle n \rangle}\|\boldsymbol{x}_{\theta}^{(i)} - \boldsymbol{x}^{(i)}\| - \mu\gamma^{\langle n \rangle}\|\boldsymbol{x}^{*\langle n \rangle} - \boldsymbol{x}_{\theta}^{(i)}\| = \\ z^{(i)} - \mu\gamma^{\langle n \rangle}\left(\|\boldsymbol{x}_{\theta}^{(i)} - \boldsymbol{x}^{(i)}\| + \|\boldsymbol{x}_{\theta}^{(i)} - \boldsymbol{x}^{*\langle n \rangle}\|\right) = \\ z^{(i)} - \mu\gamma^{\langle n \rangle}\left((1-a)\|\boldsymbol{x}^{*\langle n \rangle} - \boldsymbol{x}^{(i)}\| + a\|\boldsymbol{x}^{*\langle n \rangle} - \boldsymbol{x}^{(i)}\|\right) = \\ z^{(i)} - \mu\gamma^{\langle n \rangle}\|\boldsymbol{x}^{*\langle n \rangle} - \boldsymbol{x}^{(i)}\|, \end{aligned}$$

which contradicts equation (21). Then, (23) is false, while (22) holds true, implying that point $\boldsymbol{x}_{\theta}^{(i)}$ satisfies constraint (17b), hence proving the result. □

### *4.1. Convergence of SMGO algorithm*

We next show that after finite iterations, an $\epsilon$-suboptimal point is obtained, as required by Problem 1. For simplicity, we consider the following technical assumption, which in practice can be replaced by a vertex estimation mechanism presented in Section 5.1. Let us denote with $V \in \mathbb{N}$ the number of vertices of the polytope $\mathcal{X}$, and with $\boldsymbol{c}^{(v)}$ the $v$-th vertex.

**Assumption 3.** *The starting data set $\boldsymbol{X}^{\langle n_0 \rangle}$ contains all the vertices of $\mathcal{X}$: $\boldsymbol{c}^{(v)} \in \boldsymbol{X}^{\langle n_0 \rangle}, \forall v \in [1, \ldots, V]$.*

We start by proving four lemmas that are instrumental to show SMGO convergence.

In the following, for a point $\boldsymbol{x}$ and positive scalar $r$, denote with $\mathcal{B}(\boldsymbol{x}, r)$ the closed hyper-ball:

$$\mathcal{B}(\boldsymbol{x}, r) = \{\boldsymbol{y} : \|\boldsymbol{y} - \boldsymbol{x}\| \leq r\}$$

**Lemma 2.** *Mode $\theta$ will fail after a finite number of iterations.*

*Proof.* At iteration $n$, denote with $\mathcal{G}^{\langle n \rangle}$ the subset of the search space where a candidate test point is accepted to perform an exploitation, i.e.,

$$\mathcal{G}^{\langle n \rangle} \triangleq \left\{\boldsymbol{x} \in \mathcal{X} \ : \ \underline{z}^{\langle n \rangle}(\boldsymbol{x}) < z^{*\langle n \rangle} - \alpha\gamma^{\langle n \rangle}\right\}.$$

When a candidate point inside this set is chosen as $\boldsymbol{x}^{(n+1)}$ in Mode $\theta$, two cases arise:

1. The new sample does not change the current best point $\boldsymbol{x}^{*\langle n \rangle}$. In this case, $z^{(n+1)} \geq z^{*\langle n \rangle}$.

Denote as $\mathcal{B}(\boldsymbol{x}^{(n+1)}, r_\theta)$, the hyper-ball centered at $\boldsymbol{x}^{(n+1)}$ with finite radius

$$r_\theta \triangleq \frac{z^{(n+1)} - \left(z^{*\langle n\rangle} - \alpha\gamma^{\langle n\rangle}\right)}{\mu\gamma^{\langle n+1\rangle}}.$$

It follows that $\forall \boldsymbol{x} \in \mathcal{B}(\boldsymbol{x}^{(n+1)}, r_\theta)$

$$\underline{z}^{\langle n+1\rangle}(\boldsymbol{x}) \geq z^{*\langle n\rangle} - \alpha\gamma^{\langle n\rangle}.$$

Therefore all the points inside the hyper-ball are not eligible for exploitation. Hence, we have:

$$\mathcal{G}^{\langle n+1\rangle} = \mathcal{G}^{\langle n\rangle} \setminus \mathcal{B}(\boldsymbol{x}^{(n+1)}, r_\theta)$$

i.e., $\mathcal{G}$ diminishes by a finite amount, which is valid even when $\gamma^{\langle n\rangle}$ updates, because $\gamma^{\langle n\rangle}$ is bounded by $\gamma_o$.

2. The new sample replaces the current best point $\boldsymbol{x}^{*\langle n\rangle}$. In this case $z^{(n+1)} < z^{*\langle n\rangle}$ and therefore $\mathcal{G}^{\langle n+1\rangle} \subset \mathcal{G}^{\langle n\rangle}$ due to the new threshold. Moreover, the hyper-ball $\mathcal{B}(\boldsymbol{x}^{(n+1)}, r_\theta)$, with $r_\theta = \dfrac{\alpha}{\mu}$ around the new sample is also removed from $\mathcal{G}$.

Hence, $\mathcal{G}^{\langle n\rangle} = \varnothing$ after finite iterations, and Mode $\theta$ will fail, proving the lemma. □

**Lemma 3.** *Let Assumption 3 hold and consider any unsampled point $\hat{\boldsymbol{x}}$ in the interior of $\mathcal{X}$ and any sample $\boldsymbol{x}^{(i)} \in \boldsymbol{X}^{\langle n\rangle}$. Consider the open half-space:*

$$\mathcal{Q} = \left\{\boldsymbol{x} \in \mathcal{X} \ : \ (\boldsymbol{x} - \boldsymbol{x}^{(i)})^\top(\hat{\boldsymbol{x}} - \boldsymbol{x}^{(i)}) > 0\right\}.$$

*Then, there exists at least one sample $\boldsymbol{x}^{(j)} \in \boldsymbol{X}^{\langle n\rangle} \cap \mathcal{Q}$, $i \neq j$.*

*Proof.* Let us assume, for the purpose of contradiction, that the lemma does not hold. That is, there is no sample in the open half-space $\mathcal{Q}$. Due to Assumption 3, this would also apply to all vertices $\boldsymbol{c}^{(v)}$, $v = 1, \ldots, V$ of $\mathcal{X}$. Therefore, we would have:

$$(\hat{\boldsymbol{x}} - \boldsymbol{x}^{(i)})^\top(\boldsymbol{c}^{(v)} - \boldsymbol{x}^{(i)}) \leq 0, \forall v = 1, \ldots, V. \quad (24)$$

Two cases could then arise:

1. Inequality in (24) applies strictly for at least one vertex $\boldsymbol{c}^{(v)}$. In this case, any such vertex $\boldsymbol{c}^{(v)}$ and $\hat{\boldsymbol{x}}$ would reside on opposite sides of the hyper-plane

$$\mathcal{P} \triangleq \left\{\boldsymbol{x} \ : \ (\hat{\boldsymbol{x}} - \boldsymbol{x}^{(i)})^\top(\boldsymbol{x} - \boldsymbol{x}^{(i)}) = 0\right\}.$$

However, consider any $\boldsymbol{w} \in \mathcal{X}$. Since the latter is a convex polytope, we can write

$$\boldsymbol{w} = \sum_{v=1}^{V} b_v \boldsymbol{c}^{(v)} \quad (25)$$

with $0 \leq b_v \leq 1$, $v = 1, \ldots, V$, and $\sum_{v=1}^{V} b_v = 1$. Hence, it would apply that

$$\forall \boldsymbol{w} \in \mathcal{X}, \ (\hat{\boldsymbol{x}} - \boldsymbol{x}^{(i)})^\top(\boldsymbol{w} - \boldsymbol{x}^{(i)}) =$$
$$(\hat{\boldsymbol{x}} - \boldsymbol{x}^{(i)})^\top \left(\sum_{v=1}^{V} b_v \boldsymbol{c}^{(v)} - \sum_{v=1}^{V} b_v \boldsymbol{x}^{(i)}\right) =$$
$$\sum_{v=1}^{V} b_v (\hat{\boldsymbol{x}} - \boldsymbol{x}^{(i)})^\top(\boldsymbol{c}^{(v)} - \boldsymbol{x}^{(i)}) < 0,$$

which would mean that all $\boldsymbol{w} \in \mathcal{X}$ reside on the opposite side of $\mathcal{P}$ with respect to $\hat{\boldsymbol{x}}$ . This can only happen if $\hat{\boldsymbol{x}} \notin \mathcal{X}$, falling in a contradiction with the assumptions.

2. Equality in (24) applies for all $\boldsymbol{c}^{(v)}$, $v = 1, \ldots, V$. This would mean that all vertices (and consequently, all $\boldsymbol{w} \in \mathcal{X}$) belong to the hyper-plane $\mathcal{P}$ containing $\boldsymbol{x}^{(i)}$. In turn, this would imply that either $\hat{\boldsymbol{x}} \notin \mathcal{X}$ (if $\|\hat{\boldsymbol{x}} - \boldsymbol{x}^{(i)}\| > 0$), or $\hat{\boldsymbol{x}} = \boldsymbol{x}^{(i)}$, both of which fall in contradiction with the assumptions.

Since both cases 1. and 2. lead to contradiction, the claim of the lemma must be true, thus completing the proof. □

**Lemma 4.** *Consider any unsampled point $\hat{\boldsymbol{x}}$ and any radius $r$ such that $\boldsymbol{X}^{\langle n\rangle} \cap \mathcal{B}(\hat{\boldsymbol{x}}, r) = \varnothing$. Then, given Assumption 2 and a Lipschitz constant estimate $\gamma^{\langle n\rangle}$ obtained with (7)-(8), it applies that*

$$\lambda^{\langle n\rangle}(\hat{\boldsymbol{x}}) \geq 2(\mu - 1)\gamma^{\langle n\rangle} r. \quad (26)$$

*Proof.* Consider the point $\hat{\boldsymbol{x}}$ and denote with $(\boldsymbol{x}^{(a)}, z^{(a)})$, $(\boldsymbol{x}^{(b)}, z^{(b)}) \in \boldsymbol{X}^{\langle n\rangle}$ the data pairs that determine the upper and lower bounds $\overline{z}^{\langle n\rangle}(\hat{\boldsymbol{x}})$, $\underline{z}^{\langle n\rangle}(\hat{\boldsymbol{x}})$, respectively, i.e:

$$\overline{z}^{\langle n\rangle}(\hat{\boldsymbol{x}}) = \min_{k=1\ldots n} \left(z^{(k)} + \mu\gamma^{\langle n\rangle}\|\hat{\boldsymbol{x}} - \boldsymbol{x}^{(k)}\|\right) =$$
$$z^{(a)} + \mu\gamma^{\langle n\rangle}\|\hat{\boldsymbol{x}} - \boldsymbol{x}^{(a)}\|$$

$$\underline{z}^{\langle n\rangle}(\hat{\boldsymbol{x}}) = \max_{k=1,\dots,n}\left(z^{(k)} - \mu\gamma^{\langle n\rangle}\|\hat{\boldsymbol{x}} - \boldsymbol{x}^{(k)}\|\right) = z^{(b)} - \mu\gamma^{\langle n\rangle}\|\hat{\boldsymbol{x}} - \boldsymbol{x}^{(b)}\|.$$

Then, considering that $\boldsymbol{X}^{\langle n\rangle} \cap \mathcal{B}(\hat{\boldsymbol{x}}, r) = \varnothing$, that $(\mu - 1) > 0$, and taking into account (7)-(8) we have:

$$\begin{aligned}
\lambda^{\langle n\rangle}(\hat{\boldsymbol{x}}) &= \bar{z}^{\langle n\rangle}(\hat{\boldsymbol{x}}) - \underline{z}^{\langle n\rangle}(\hat{\boldsymbol{x}}) \\
&= z^{(a)} - z^{(b)} + \\
&\quad \mu\gamma^{\langle n\rangle}\left(\|\hat{\boldsymbol{x}} - \boldsymbol{x}^{(a)}\| + \|\hat{\boldsymbol{x}} - \boldsymbol{x}^{(b)}\|\right) \\
&\geq -\gamma^{\langle n\rangle}\|\boldsymbol{x}^{(a)} - \boldsymbol{x}^{(b)}\| + \\
&\quad \mu\gamma^{\langle n\rangle}\left(\|\hat{\boldsymbol{x}} - \boldsymbol{x}^{(a)}\| + \|\hat{\boldsymbol{x}} - \boldsymbol{x}^{(b)}\|\right) \\
&\geq -\gamma^{\langle n\rangle}\left(\|\hat{\boldsymbol{x}} - \boldsymbol{x}^{(a)}\| + \|\hat{\boldsymbol{x}} - \boldsymbol{x}^{(b)}\|\right) + \\
&\quad \mu\gamma^{\langle n\rangle}\left(\|\hat{\boldsymbol{x}} - \boldsymbol{x}^{(a)}\| + \|\hat{\boldsymbol{x}} - \boldsymbol{x}^{(b)}\|\right) \\
&\geq 2\,(\mu - 1)\,\gamma^{\langle n\rangle} r
\end{aligned}$$

which proves the result. □

Note that in Lemma 4 we are not assuming that the estimate $\gamma^{\langle n\rangle}$ is larger than $\gamma_o$: the result follows by how the estimate is computed, combined with the use of $\mu > 1$.

**Lemma 5.** *Let Assumption 3 hold, and assume that Mode $\psi$ is undertaken infinitely often as $n \to +\infty$. Then, for any $\hat{\boldsymbol{x}} \in \mathcal{X}$ and any $\sigma > 0$, $\exists n_\sigma < \infty$ such that*

$$\min_{\boldsymbol{x}^{(i)} \in \boldsymbol{X}^{\langle n_\sigma\rangle}} \|\boldsymbol{x}^{(i)} - \hat{\boldsymbol{x}}\| < \sigma.$$

*Proof.* Consider any point $\hat{\boldsymbol{x}} \in \mathcal{X}$. If $\hat{\boldsymbol{x}} \in \boldsymbol{X}^{\langle n\rangle}$ for some $n < \infty$, we have $\min_{\boldsymbol{x}^{(i)} \in \boldsymbol{X}^{\langle n\rangle}} \|\boldsymbol{x}^{(i)} - \hat{\boldsymbol{x}}\| = 0$ and the claim is trivially proven. Consider then the case $\hat{\boldsymbol{x}} \notin \boldsymbol{X}^{\langle n\rangle}$, $\forall n \in [1, +\infty)$. Pick the nearest sample

$$\bar{\boldsymbol{x}}^{\langle n\rangle} = \min_{\boldsymbol{x}^{(i)} \in \boldsymbol{X}^{\langle n\rangle}} \|\boldsymbol{x}^{(i)} - \hat{\boldsymbol{x}}\| \tag{27}$$

and consider the open half-space:

$$\mathcal{Q} = \left\{\boldsymbol{x} \in \mathcal{X} \;:\; (\boldsymbol{x} - \bar{\boldsymbol{x}}^{\langle n\rangle})^\top(\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}^{\langle n\rangle}) > 0\right\}.$$

Now, we select sample $\tilde{\boldsymbol{x}}^{\langle n\rangle}$

$$\tilde{\boldsymbol{x}}^{\langle n\rangle} = \operatorname*{arg\,min}_{\boldsymbol{x}^{(j)} \in \boldsymbol{X}^{\langle n\rangle} \cap \mathcal{Q}} \|\boldsymbol{x}^{(j)} - \bar{\boldsymbol{x}}^{\langle n\rangle}\|. \tag{28}$$

Such a sample is always guaranteed to exist, in force of Lemma 3. We note that due to our selection criterion for $\tilde{\boldsymbol{x}}^{\langle n\rangle}$, the set

$$\mathcal{H} = \left\{\boldsymbol{x} \in \mathcal{Q} \;:\; \|\boldsymbol{x} - \bar{\boldsymbol{x}}^{\langle n\rangle}\| < \|\tilde{\boldsymbol{x}}^{\langle n\rangle} - \bar{\boldsymbol{x}}^{\langle n\rangle}\|\right\}$$

does not have any sample in its interior, otherwise any such sample should have been selected as $\tilde{\boldsymbol{x}}^{\langle n\rangle}$ in (28). The geometry of $\mathcal{H}$ is an open half-hyperball, with center at $\bar{\boldsymbol{x}}^{\langle n\rangle}$ and radius $\|\tilde{\boldsymbol{x}}^{\langle n\rangle} - \bar{\boldsymbol{x}}^{\langle n\rangle}\|$.

We now show that a midpoint inside $\mathcal{H}$ is going to be sampled after finite iterations. There exists at least the midpoint $\hat{\boldsymbol{m}} \triangleq \dfrac{\bar{\boldsymbol{x}}^{\langle n\rangle} + \tilde{\boldsymbol{x}}^{\langle n\rangle}}{2}$ in the interior of $\mathcal{H}$. Consider the ball $\mathcal{B}(\hat{\boldsymbol{m}}, h)$ centered at $\hat{\boldsymbol{m}}$ with radius $h$:

$$\begin{aligned} h = &\max_{a \in \mathbb{R}^+} a \\ &\text{s.t.} \\ &\mathcal{B}(\hat{\boldsymbol{m}}, a) \subset \mathcal{H} \end{aligned} \tag{29}$$

Then, recalling that there are no samples inside $\mathcal{H}$, by Lemma 4 we have:

$$\lambda^{\langle n\rangle}(\hat{\boldsymbol{m}}) \geq 2(\mu - 1)\gamma^{\langle n\rangle} h. \tag{30}$$

Furthermore, we consider the set of points not belonging to $\mathcal{H}$, whose uncertainty is larger than that of $\hat{\boldsymbol{m}}$

$$\mathcal{R}^{\langle n\rangle}(\hat{\boldsymbol{m}}) = \left\{\boldsymbol{x} \notin \mathcal{H} \;:\; \lambda^{\langle n\rangle}(\boldsymbol{x}) > \lambda^{\langle n\rangle}(\hat{\boldsymbol{m}})\right\}.$$

We then sort all candidate midpoints in the set $\boldsymbol{X}_\psi^{\langle n\rangle}$, by decreasing $\lambda^{\langle n\rangle}$ values:

$$\boldsymbol{M}^{\langle n\rangle} \triangleq \left\{\boldsymbol{m}_1^{\langle n\rangle}, \dots, \boldsymbol{m}_M^{\langle n\rangle}\right\} : \lambda^{\langle n\rangle}(\boldsymbol{m}_k^{\langle n\rangle}) \geq \lambda^{\langle n\rangle}(\boldsymbol{m}_{k+1}^{\langle n\rangle})$$

where $\boldsymbol{m}_k^{\langle n\rangle} = \boldsymbol{x}_\psi^{(i,j)}$ for some $\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)} \in \boldsymbol{X}^{\langle n\rangle}$ (see (19)) and $M = \dfrac{n(n-1)}{2}$ is the total number of segments at iteration $n$. In Mode $\psi$, the first-ranking candidate $\boldsymbol{m}_1^{\langle n\rangle}$ is taken for sampling. Its uncertainty is necessarily larger than $\lambda^{\langle n\rangle}(\hat{\boldsymbol{m}})$ (otherwise $\hat{\boldsymbol{m}}$ would have been ranked higher), i.e. $\boldsymbol{m}_1^{\langle n\rangle} \in \mathcal{R}^{\langle n\rangle}(\hat{\boldsymbol{m}})$. Assume, for the purpose of contradiction, that points in $\mathcal{H}$ are never sampled.

At iteration $n + 1$, the midpoint $\boldsymbol{m}_1^{\langle n\rangle} \notin \mathcal{H}$ is thus sampled and its uncertainty $\lambda^{\langle n+1\rangle}(\boldsymbol{m}_1^{\langle n\rangle})$ becomes zero. Since this point does not belong to $\mathcal{H}$, the value of $h$ in (29) is still valid, while the uncertainty bound (30) may either be the same, or increase in case $\gamma^{\langle n+1\rangle} > \gamma^{\langle n\rangle}$ (see (8)). Moreover, for any

point $\boldsymbol{x} \in \mathcal{X}$ we have

$$\lambda^{\langle n+1 \rangle}(\boldsymbol{x}) \leq 2\mu\gamma^{\langle n+1 \rangle}\|\boldsymbol{x} - \boldsymbol{m}_1^{\langle n \rangle}\| \quad (31)$$

as shown by direct application of (9)-(11). Consider now the ball $\mathcal{B}(\boldsymbol{m}_1^{\langle n \rangle}, r_\lambda)$, where

$$r_\lambda = \left(1 - \frac{1}{\mu}\right) h. \quad (32)$$

On the basis of (30), (31) and (32) we have that:

$$\forall \boldsymbol{x} \in \mathcal{B}(\boldsymbol{m}_1^{\langle n \rangle}, r_\lambda),\ \lambda^{\langle n+1 \rangle}(\boldsymbol{x}) \leq 2\mu\gamma^{\langle n+1 \rangle} r_\lambda = 2(\mu - 1)\gamma^{\langle n+1 \rangle} h \leq \lambda^{\langle n+1 \rangle}(\hat{\boldsymbol{m}}). \quad (33)$$

Hence,

$$\mathcal{R}^{\langle n+1 \rangle}(\hat{\boldsymbol{m}}) = \mathcal{R}^{\langle n \rangle}(\hat{\boldsymbol{m}}) \backslash \left( \mathcal{R}^{\langle n \rangle}(\hat{\boldsymbol{m}}) \cap \mathcal{B}(\boldsymbol{m}_1^{\langle n \rangle}, r_\lambda) \right).$$

This means that the volume of set $\mathcal{R}^{\langle n \rangle}(\hat{\boldsymbol{m}})$ diminishes by a finite quantity at each iteration, since the value of $h$ does not depend on $n$; and becomes null after finite iterations, since Mode $\psi$ is assumed to be undertaken infinitely often. Denote with $\bar{n} - 1 < \infty$ the iteration at which this happens. Then, no midpoint outside the set $\mathcal{H}$ would have uncertainty larger than $\lambda^{\langle \bar{n} \rangle}(\hat{\boldsymbol{m}})$, so that a midpoint

$$\left\{ \boldsymbol{x}_\psi^{(i,j)} \in \boldsymbol{X}_\psi^{\langle \bar{n} \rangle} \cap \mathcal{H} \ :\ \lambda^{\langle \bar{n} \rangle}(\boldsymbol{x}_\psi^{(i,j)}) \geq \lambda^{\langle \bar{n} \rangle}(\hat{\boldsymbol{m}}) \right\}$$

will be ranked first in $\boldsymbol{M}^{\langle \bar{n}-1 \rangle}$ and sampled at iteration $\bar{n}$, thus falling in contradiction.
We thus demonstrated that, in finite iterations, a point inside $\mathcal{H}$ is sampled. Moreover, denoting such a point as $\boldsymbol{x}^{(\bar{n})}$, by construction we have

$$\|\boldsymbol{x}^{(\bar{n})} - \bar{\boldsymbol{x}}^{\langle n \rangle}\| < \|\tilde{\boldsymbol{x}}^{\langle n \rangle} - \bar{\boldsymbol{x}}^{\langle n \rangle}\|. \quad (34)$$

Then, two cases may occur: either $\boldsymbol{x}^{(\bar{n})}$ is closer to $\hat{\boldsymbol{x}}$ than $\bar{\boldsymbol{x}}^{\langle n \rangle}$, or not. In the former case, repeating the process from (27) we'll have that $\bar{\boldsymbol{x}}^{\langle \bar{n} \rangle} = \boldsymbol{x}^{(\bar{n})}$ and $\tilde{\boldsymbol{x}}^{\langle \bar{n} \rangle} = \bar{\boldsymbol{x}}^{\langle n \rangle}$. In the latter case, $\bar{\boldsymbol{x}}^{\langle \bar{n} \rangle} = \bar{\boldsymbol{x}}^{\langle n \rangle}$ and $\tilde{\boldsymbol{x}}^{\langle \bar{n} \rangle} = \boldsymbol{x}^{(\bar{n})}$. In both cases, the radius of the half-hyper-ball $\mathcal{H}$ shrinks because of (34). By induction we thus have:

$$\lim_{n \to \infty} \|\tilde{\boldsymbol{x}}^{\langle n \rangle} - \bar{\boldsymbol{x}}^{\langle n \rangle}\| = 0. \quad (35)$$

Next, still considering $\bar{\boldsymbol{x}}^{\langle n \rangle}$ as the nearest sample to $\hat{\boldsymbol{x}}$ (27), we are going to show that there exists a radius $a > 0$ such that

$$\|\boldsymbol{x} - \hat{\boldsymbol{x}}\| < \|\bar{\boldsymbol{x}}^{\langle n \rangle} - \hat{\boldsymbol{x}}\|, \forall \boldsymbol{x} \in \mathcal{Q} \cap \mathcal{B}(\bar{\boldsymbol{x}}^{\langle n \rangle}, a) \quad (36)$$

To this end, consider any point $\boldsymbol{x} \in \mathcal{Q}$ and assume, for the sake of contradiction, that (36) never holds. This would mean that the nearest point to $\hat{\boldsymbol{x}}$ on the segment connecting $\bar{\boldsymbol{x}}^{\langle n \rangle}$ to $\boldsymbol{x}$ is always $\bar{\boldsymbol{x}}^{\langle n \rangle}$, no matter how close the two points are. In turn, this would imply that:

$$\frac{d}{da}\|((1-a)\bar{\boldsymbol{x}}^{\langle n \rangle} + a\boldsymbol{x}) - \hat{\boldsymbol{x}}\|\Big|_{a=0} \geq 0, \quad (37)$$

i.e.,

$$\frac{1}{\|\bar{\boldsymbol{x}}^{\langle n \rangle} - \hat{\boldsymbol{x}}\|}(\bar{\boldsymbol{x}}^{\langle n \rangle} - \hat{\boldsymbol{x}})^\top(\boldsymbol{x} - \bar{\boldsymbol{x}}^{\langle n \rangle}) \geq 0$$
$$(\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}^{\langle n \rangle})^\top(\boldsymbol{x} - \bar{\boldsymbol{x}}^{\langle n \rangle}) \leq 0$$

which would imply that $\boldsymbol{x} \notin \mathcal{Q}$, thus falling in a contradiction. Hence, the directional derivative (37) must be negative for all $\boldsymbol{x} \in \mathcal{Q}$, meaning that there exists a scalar $a > 0$ such that in the neighborhood $\mathcal{Q} \cap \mathcal{B}(\bar{\boldsymbol{x}}^{\langle n \rangle}, a)$ the property (36) holds.

Combining (35) and (36), we obtain that, after a finite number $k$ of iterations, the following condition holds:

$$\|\bar{\boldsymbol{x}}^{\langle n+k \rangle} - \hat{\boldsymbol{x}}\| < \|\bar{\boldsymbol{x}}^{\langle n \rangle} - \hat{\boldsymbol{x}}\|.$$

This implies that

$$\lim_{n \to \infty} \|\bar{\boldsymbol{x}}^{\langle n \rangle} - \hat{\boldsymbol{x}}\| = 0,$$

and, for any $\sigma > 0$, we finally have

$$\exists n_\sigma : \min_{\boldsymbol{x}^{(i)} \in \boldsymbol{X}^{\langle n_\sigma \rangle}} \|\boldsymbol{x}^{(i)} - \hat{\boldsymbol{x}}\| < \sigma$$

thus proving the lemma. □

**Theorem 1.** *Let Assumptions 1 and 3 hold. Then,* $\forall \epsilon > 0,\ \exists n_\epsilon < \infty : z^{*\langle n_\epsilon \rangle} \leq z^* + \epsilon$.

*Proof.* Consider a point $\hat{\boldsymbol{x}} \in \mathcal{X}^*$ (see (2)) and take $\sigma = \dfrac{\epsilon}{\gamma_o}$. For any $n$, denote

$$\bar{\boldsymbol{x}}^{\langle n \rangle} = \arg \min_{\boldsymbol{x}^{(i)} \in \boldsymbol{X}^{\langle n \rangle}} \|\hat{\boldsymbol{x}} - \boldsymbol{x}^{(i)}\|$$

In virtue of Lemma 2, the exploration mode will be called infinitely often. Now, by applying Lemma 5, we have that $\exists n_\sigma < \infty : \|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}^{\langle n_\sigma \rangle}\| < \sigma = \dfrac{\epsilon}{\gamma_o}$. Then, in virtue of Assumption 1 we have:

$$f_o(\boldsymbol{x}^{*\langle n_\sigma\rangle}) - f_o(\hat{\boldsymbol{x}}) = z^{*\langle n\rangle} - z^* \leq$$
$$f_o(\bar{\boldsymbol{x}}^{\langle n_\sigma\rangle}) - f_o(\hat{\boldsymbol{x}}) \leq \gamma_o \|\hat{\boldsymbol{x}} - \bar{\boldsymbol{x}}^{\langle n_\sigma\rangle}\| < \epsilon \quad (38)$$

Thus proving the result with $n_\epsilon = n_\sigma$. □

### *4.2. Optimality gap*

Considering a run of the SMGO algorithm with $N$ iterations, we now analyze the gap between the best sampled value and the actual global minimum,

$$\delta^{\langle N\rangle} = z^{\langle N\rangle *} - f_o(x^*)$$

denoted as the *optimality gap*. In calculating such a difference, the SM-based guarantees $\underline{z}^{\langle N\rangle}(\boldsymbol{x}) \leq f_o(\boldsymbol{x}) \leq \overline{z}^{\langle N\rangle}(\boldsymbol{x})$ can be utilized in the case of a known Lipschitz constant $\gamma_o$. Hence, the upper bound of the optimality gap $\overline{\delta}^{\langle N\rangle}$ can be considered as the difference between $z^{\langle N\rangle *}$ and the minimum lower bound $\underline{z}^{\langle N\rangle}$ defined in (9) generated by data set $\boldsymbol{X}^{\langle N\rangle}$, i.e.

$$\begin{aligned}\overline{\delta}^{\langle N\rangle} &= \max_{\boldsymbol{x}\in\mathcal{X}} \left( z^{\langle N\rangle *} - \underline{z}^{\langle N\rangle}(\boldsymbol{x}) \right) \\ &= z^{\langle N\rangle *} - \min_{\boldsymbol{x}\in\mathcal{X}} \underline{z}^{\langle N\rangle}(\boldsymbol{x}) \end{aligned} \quad (39)$$

The calculation of $\min_{\boldsymbol{x}\in\mathcal{X}} \underline{z}^{\langle N\rangle}(\boldsymbol{x})$ in (39) entails calculation of the hyperbolic Voronoi cells and of the lower bounds at their vertices. Denoting with $\boldsymbol{V}^{\langle N\rangle}$ the set of such vertices generated from $\boldsymbol{X}^{\langle N\rangle}$, (39) can be expressed as

$$\overline{\delta}^{\langle N\rangle} = z^{\langle N\rangle *} - \min_{\boldsymbol{v}\in\boldsymbol{V}^{\langle N\rangle}} \underline{z}^{\langle N\rangle}(\boldsymbol{v}). \quad (40)$$

The acquisition of HVD vertices using the samples can be approached as described in [24], which is referred for more information. The calculation of HVD cells and vertices entail computations with exponential complexity w.r.t. $D$ [24]. However, an interested user can obtain an upper bound on the optimality gap by carrying out (40).

## 5. Computational Aspects

This section presents an analysis of the computational aspects of the proposed algorithm and possible improvements. The first aspect discussed is on ensuring the coverage of $\mathcal{X}$, followed by a discussion on the computational complexity.

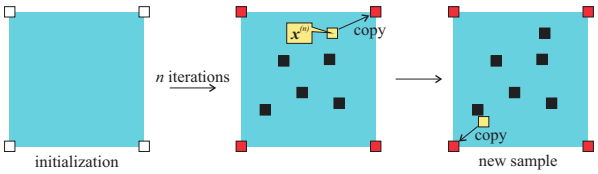


Figure 3: Corner mirroring of nearest sample point

### *5.1. On the coverage of the search set*

The selection method for candidate points described in Section 3 allows SMGO to only explore the volume inside the convex hull of the sampled points in $\boldsymbol{X}^{\langle n_0\rangle}$. Hence, in order to ensure exploration throughout all the search set $\mathcal{X}$, all its vertices $\boldsymbol{c}^{(v)}$ should also be considered in the candidate points generation, as stated in Assumption 3. One approach is to first evaluate $f_o$ on all $\boldsymbol{c}^{(v)}$ before actually starting with exploitation/exploration. However, in the case of a hyperrectangle, this entails an exponential number of long function calls w.r.t. dimensionality $D$.

To address this issue, a simple approach is proposed in order to ensure coverage in $\mathcal{X}$ without explicitly calling $f_o$ to evaluate the corners. Moreover, with this approach Assumption 3 is not needed anymore for the theoretical results to hold. At each iteration $n$, the midpoints between any sample and all the vertices of $\mathcal{X}$ are included as candidate exploration samples, even if the vertices do not belong to $\boldsymbol{X}^{\langle n\rangle}$. Moreover, for the purpose of computing the uncertainty of each midpoint, the cost function value $f_o(\boldsymbol{c}^{(v)})$ at the $v$-th vertex $\boldsymbol{c}^{(v)}$ is estimated as:

$$f_o(\boldsymbol{c}^{(v)}) \approx z^{(w)}, \text{ where } w = \arg\min_{k=1,\ldots,n} \|\boldsymbol{x}^{(k)} - \boldsymbol{c}^{(v)}\| \quad (41)$$

i.e., equal to the cost function value at the nearest sampled point. The concept of this strategy is shown in Fig. 3. Thanks to Assumption 1 and Lemma 5, with an increasing number of sample points in $\mathcal{X}$ the estimate (41) approximates more and more accurately the true function value at the vertex.

### *5.2. On the computational burden*

There are three main routines within an iteration of SMGO: Lipschitz constant update, exploitation by Mode $\theta$ routine, and eventually, if the expected improvement threshold is not achieved, exploration by Mode $\psi$. The worst-case SMGO complexity

analysis assumes that each iteration eventually proceeds to Mode $\psi$.

The first routine compares the incoming $n$-th sample with $n-1$ existing ones, resulting in $\mathcal{O}(n)$ complexity.

The Modes $\theta$ and $\psi$ can be designed as iterative routines as well to significantly improve the computational burden, as discussed in the following.

The exploitation Mode $\theta$ comprises the selection of $n-1$ candidate points $\boldsymbol{x}_\theta^{(i)}$ for each segment $l_{n^*,i}$ from $\boldsymbol{x}^{*\langle n\rangle}$ to all the other sampled points according to (15)-(16), and the calculation of $\underline{z}^{\langle n\rangle}$ for each $\boldsymbol{x}_\theta^{(i)}$ using (9). Both are $\mathcal{O}(n)$ operations, leading to a compounded complexity of $\mathcal{O}(n^2)$. However, when $\gamma^{\langle n\rangle}$ and $\boldsymbol{x}^{*\langle n\rangle}$ do not change from iteration $n-1$ to $n$, a cache can store $\boldsymbol{x}_\theta^{(i)}$ locations and corresponding $\underline{z}^{\langle n\rangle}$. From (9), it is convenient to save $z^{(\tilde{k})}$ and $\tilde{\Delta}^{(i)} \triangleq \mu\gamma^{\langle n\rangle}\|\boldsymbol{x}_\theta^{(i)} - \boldsymbol{x}^{(\tilde{k})}\|$, where

$$\tilde{k} = \underset{k\in[1\dots n]}{\arg\max} \left( z^{(k)} - \mu\gamma^{\langle n\rangle}\|\boldsymbol{x}_\theta^{(i)} - \boldsymbol{x}^{(k)}\| \right).$$

In practice, information are saved[1] about the hypercone that actually generated $\underline{z}^{\langle n\rangle}$ for each $\boldsymbol{x}_\theta^{(i)}$. As long as $\boldsymbol{x}^{*\langle n\rangle}$ stays the same, previously existing $\boldsymbol{x}_\theta^{(i)}$ locations do not need to be computed again.

Furthermore, calculating $\underline{z}^{\langle n\rangle}$ for an existing exploitation point $\boldsymbol{x}_\theta^{(i)}$ reduces to

$$\underline{z}^{\langle n\rangle}(\boldsymbol{x}_\theta^{(i)}) = \max\left(\underline{z}^{\langle n-1\rangle}(\boldsymbol{x}_\theta^{(i)}), \underline{z}_{new}^{(i)}\right) \tag{42}$$

where

$$\underline{z}_{new}^{(i)} = z^{(n)} - \mu\gamma^{\langle n\rangle}\|\boldsymbol{x}_\theta^{(i)} - \boldsymbol{x}^{(n)}\|$$

for an incoming sample $(\boldsymbol{x}^{(n)}, z^{(n)})$.

Note that when $\underline{z}^{\langle n\rangle}(\boldsymbol{x}_\theta^{(i)})$ changes due to (42), $z^{(\tilde{k})} = z^{(n)}$ and $\tilde{\Delta}^{(i)} = \mu\gamma^{\langle n\rangle}\|\boldsymbol{x}_\theta^{(i)} - \boldsymbol{x}^{(n)}\|$. As a result, computing $\underline{z}^{\langle n\rangle}$ for all existing candidate points is $\mathcal{O}(n)$, i.e. $\mathcal{O}(1)$ for each. Now, consider the situation when $\gamma^{\langle n\rangle}$ is updated, assuming that $\boldsymbol{x}^{*\langle n\rangle}$ and $\tilde{k}$ do not change. This implies rescaling $\tilde{\Delta}^{(i)}$ such that

$$\tilde{\Delta}_{\langle n\rangle}^{(i)} = \frac{\gamma^{\langle n\rangle}}{\gamma^{\langle n-1\rangle}}\tilde{\Delta}_{\langle n-1\rangle}^{(i)} \tag{43}$$

[1]It can be understood that $z^{(\tilde{k})}$ is the value at the tip, while $\tilde{\Delta}^{(i)}$ is the height of the hypercone.

and updating $\underline{z}^{\langle n\rangle}$ with the new sample $(\boldsymbol{x}^{(n)}, z^{(n)})$ using (42), still resulting in $\mathcal{O}(1)$ per candidate point. This implies a Mode $\theta$ aggregate complexity of $\mathcal{O}(n)$.

Finally, one new candidate point $\boldsymbol{x}_\theta^{(n)}$ is evaluated using (9), which is $\mathcal{O}(n)$. Hence, as long as $\boldsymbol{x}^{*\langle n\rangle}$ does not change, the introduction of a cache reduces Mode $\theta$ to $\mathcal{O}(n)$. However, when $\boldsymbol{x}^{*\langle n\rangle}$ changes, i.e. $\boldsymbol{x}^{*\langle n\rangle} = \boldsymbol{x}^{(n)}$, computing all $\boldsymbol{x}_\theta^{(i)}$ locations must be repeated from the start, leading to $\mathcal{O}(n^2)$ worst-case complexity for Mode $\theta$ as stated before.

As defined, Mode $\psi$ generates $\dfrac{n(n-1)}{2}$ candidate points (midpoints among all existing samples) at each iteration. Furthermore, the calculation of $\lambda^{\langle n\rangle}$ for each candidate point is $\mathcal{O}(n)$, resulting in $\mathcal{O}(n^3)$ for each exploration iteration.

Note that behavior in Mode $\psi$ does not depend on the current best point, because $n$ new segments are added to the search set at iteration $n$. A cache solution can be introduced also in this mode, updating $\lambda^{\langle n\rangle}$-related values iteratively.

For each existing $\boldsymbol{x}_\psi^{(i,j)}$, the values to be saved are $z^{(\hat{k})}$ and $\hat{\Delta}^{(i,j)}$ (for $\overline{z}^{\langle n\rangle}$), and $z^{(\check{k})}$ and $\check{\Delta}^{(i,j)}$ (for $\underline{z}^{\langle n\rangle}$), where $\hat{k}$ and $\check{k}$ are defined as

$$\hat{k} = \underset{k\in[1\dots n]}{\arg\min} \left( z^{(k)} + \mu\gamma^{\langle n\rangle}\|\boldsymbol{x}_\psi^{(i,j)} - \boldsymbol{x}^{(k)}\| \right),$$

$$\check{k} = \underset{k\in[1\dots n]}{\arg\max} \left( z^{(k)} - \mu\gamma^{\langle n\rangle}\|\boldsymbol{x}_\psi^{(i,j)} - \boldsymbol{x}^{(k)}\| \right).$$

The iterative updates are performed using eq. (42) and (43). This then results in $\mathcal{O}(1)$ for each existing midpoint, and $\mathcal{O}(n)$ for each new midpoint generated by new segments from incoming $\boldsymbol{x}^{(n)}$ to the $n-1$ existing points. This results in $\mathcal{O}(n^2)$ for Mode $\psi$, and in turn, a $\mathcal{O}(n^2)$ worst-case complexity for SMGO.

**Remark 1.** *Note that when $\gamma^{\langle n\rangle}$ is updated (which always implies a growth of its value), the proposed iterative implementation might result in more conservative bounds $\overline{z}^{\langle n\rangle}$ and $\underline{z}^{\langle n\rangle}$ than if repeatedly recalculated at every iteration, i.e. lower $\underline{z}^{\langle n\rangle}$ for Mode $\theta$, and larger $\lambda^{\langle n\rangle}$ in Mode $\psi$. This occurs because with higher $\gamma^{\langle n\rangle}$-values the HVD layout increasingly approaches that of a non-hyperbolic Voronoi tessellation, i.e. the cones providing the tightest bounds at any $\boldsymbol{x}_\psi^{(i,j)}$ could change when*

*$\gamma^{\langle n+1\rangle} \gg \gamma^{\langle n\rangle}$, and $z^{\langle \hat{k}\rangle}$ and $z^{\langle \check{k}\rangle}$ would tend to the nearest sample w.r.t. said $\boldsymbol{x}_{\psi}^{(i,j)}$. These effects are not accounted for by the iterative update procedure, which assumes that the bound-generating cones remain the same. However, the theoretical properties of SMGO are still valid with the iterative implementation. In fact, all the arguments for Lemma 2 are valid also with iteratively-computed lower bounds. Regarding the exploration mode, inequality (26) in Lemma 4 is consistent with the more conservative iteratively-computed $\gamma^{\langle n\rangle}$. Furthermore, when sampling a new point $\boldsymbol{m}_1^{\langle n\rangle}$, for which $\lambda^{\langle n+1\rangle}(\boldsymbol{m}_1^{\langle n\rangle})$ is exactly computed, (33) still holds. Hence, all arguments laid out in Lemma 5 still hold.*

## 6. Performance Test Results

In this section, the proposed SMGO algorithm is evaluated on well-known benchmarks, usually employed in black-box optimization, comparing its performance with representative Lipschitz-based methods: DIRECT and AdaLIPO. In addition, the Bayesian optimization approach (BayesianO) is also considered in the tests. The results are discussed in terms of iteration-based optimization performance and computational times. All scripts used for this section are available at `https://github.com/lorenzosabugjr/smgo`.

### *6.1. Test parameters*

Seven test functions of varying structure and dimensions are chosen for this test, covering a variety of characteristics relevant to black-box optimization benchmarks. Their search bounds, optimal values and relevant features are summarized in Table 1. A reasonable assumption in real-life applications is that the black-box function has no available optimal value. Hence, each algorithm is given a budget of $N = 500$ long function evaluations for each optimization run. The optimal results after the evaluation budget are then measured and compared among the different algorithms, in what is referred as a fixed-cost/budget comparison [25]. All computations were performed in MATLAB 2020b on a system with AMD Ryzen 9 3900X (3.80 GHz) and 32 GB RAM.

To assess the sensitivity of each algorithm to different initial information and randomized decisions, 100 independent runs (trials) were performed. For each trial, the same randomly-generated starting point is given to AdaLIPO, SMGO, and BayesianO. On the other hand, DIRECT considers a fixed set of search points, not allowing to randomize the starting samples.

SMGO is implemented as described in Algorithm 1 with parameters $\alpha = 0.001$ and $\mu = 1.025$. AdaLIPO is implemented as described in [18] with parameter $p = 0.1$. DIRECT is implemented as described in [14], not requiring any tuning parameter. Finally, the algorithm available in the Matlab Statistics and Machine Learning Toolbox is employed for the BayesianO implementation, using the provided default parameters.

### *6.2. Iteration-based performance*

The evolution of the best sample $z^{*\langle n\rangle}$ along the iterations ($n \in [50, 500]$) for each algorithm is shown in Figs. 4 and 5 for the Rosenbrock and Deb's functions, respectively. The graphs show the distribution of the best sampled cost for the 100 independent runs. Note that the results of the DIRECT algorithm do not show dispersion due to the fact that it is a deterministic search method. DIRECT follows a batch sampling strategy, hence the graphs show for each $n$ value, the best results for the previously-sampled batches.

The Rosenbrock function is a standard test case for gradient-based algorithms, with a unique global minimum (unimodal) for $D = 2$; however it has 2 minima for $D = 4 \sim 30$ [26]. The location of $\boldsymbol{x}^*$ is at $(1, 1, \ldots, 1)$, hence the classical search bounds $-10 \leq x_i \leq 10$ would be advantageous for DIRECT, which invariably samples the center of the search hyper-box. For this reason, the search bounds are changed to move $\boldsymbol{x}^*$ away from the center.

As seen in Fig. 4, DIRECT has only improved marginally the initial sampled cost after the 500 iterations. Also AdaLIPO achieved marginal improvements after 500 iterations. This might be associated to the completely randomized exploration and exploitation decisions. On the other hand, both SMGO and BayesianO achieved large improvements during the first 100 iterations, while slight enhancements are observed during the final 400 iterations. SMGO exhibits a wider distribution among different runs and, on average, achieves a better result after 500 iterations.

In the case of Deb's function #1, as shown in Fig. 5, the presence of numerous global minima causes a bad performance for DIRECT, which barely improved its initial best cost. On the other

| Description | Function definition $f(\boldsymbol{x})$ | Bounds | $z^*$ | Features |
|---|---|---|---|---|
| Rosenbrock | $\sum_{i=1}^{D}[100(x_{i+1}-x_i^2)^2+(1-x_i)^2]$ | $-40 \leq x_i \leq 5$ | 0.0 | unimodal, non-separable |
| Styblinski-Tang | $\frac{1}{2}\sum_{i=1}^{D}\left(x_i^4-16x_i^2+5x_i\right)$ | $-5 \leq x_i \leq 5$ | $-39.166D$ | multimodal, separable |
| Deb's #1 | $\frac{1}{D}\sum_{i=1}^{D}\sin^6(5\pi x_i)$ | $-1 \leq x_i \leq 1$ | $-1.0$ | multimodal, separable, numerous global minima |
| Deb's #2 | $\frac{1}{D}\sum_{i=1}^{D}\sin^6[5\pi(x_i^{3/4}-0.05)]$ | $0 \leq x_i \leq 150$ | $-1.0$ | multimodal, separable, numerous global minima |
| Schwefel | $-\sum_{i=1}^{D} x_i \sin\left(\sqrt{\lvert x_i\rvert}\right)$ | $-500 \leq x_i \leq 500$ | $-418.982D$ | multimodal, separable, numerous local minima |
| Salomon | $1-\cos\left(2\pi\sqrt{\sum_{i=1}^{D}x_i^2}\right)+0.1\sqrt{\sum_{i=1}^{D}x_i^2}$ | $-40 \leq x_i \leq 70$ | 0 | multimodal, non-separable |
| Brown | $\sum_{i=1}^{D-1}(x_i^2)^{(x_{i+1}^2+1)}+(x_{i+1}^2)^{(x_i^2+1)}$ | $-1 \leq x_i \leq 4$ | 0 | multimodal, non-separable |

Table 1: Test functions used for comparative tests

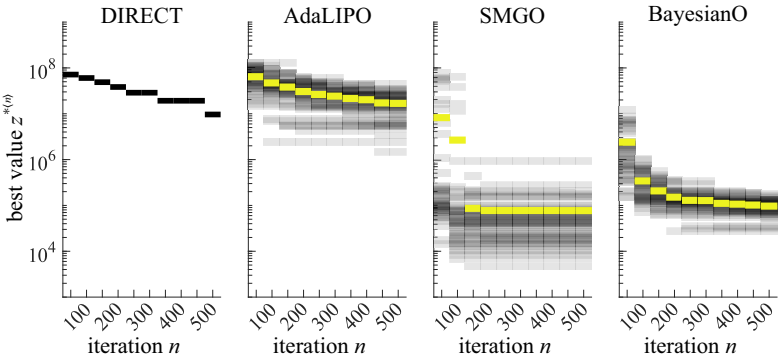


Figure 4: Optimal value distribution w.r.t. iterations, 10D Rosenbrock function (log scale)

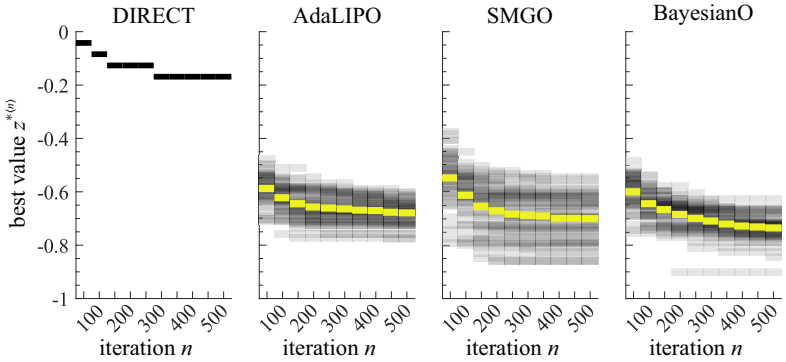


Figure 5: Optimal value distribution w.r.t. iterations, 10D Deb's function #1

hand, AdaLIPO, BayesianO, and SMGO have resulted in similar results, achieving very similar average costs after the iteration limit, while BayesianO shows more concentrated values among runs. In this case the randomized mechanism of AdaLIPO allows it to discover low-cost regions to improve the best sample. Furthermore, Bayesian's usage of a response surface, and SMGO's geometry-based candidate points derivation have become useful in looking for low-cost regions in the search space.

A summary (mean best values) of the algorithms' outcomes after 500 iterations are shown in Table 2. Note that the results tabulated for DIRECT were achieved at iterations which are a bit larger than the pre-defined maximum $N = 500$, owing to its batch-based samples generation. In most test cases, BayesianO achieved the best average results, while SMGO and DIRECT attained the best average performance in 2 cases each. On the other hand, AdaLIPO achieved fair results, even if it ranked second-place or worse for most of the tested functions.

Furthermore, for each test function (with the same starting points across all algorithms), the number of trials (out of 100) resulting in best results for each respective algorithm is shown in Table 3. As with the average performance, BayesianO achieved the most best trials in 9 out of the 13 tested functions. Furthermore, in Deb's #1 (5D), Schwefel (5D and 10D), and Salomon (5D and 10D), it resulted in best results in almost all of the given trials. On the other hand, SMGO had the most best results in Rosenbrock and Deb's #2 (10D), and had the second largest count of best trials in Deb's #1 (10D) and Deb's #2 (5D). AdaLIPO registered the second count of best trials in Deb's #2 (10D), and had no best trials for 4 functions. Lastly, DIRECT had no best trials for 8 out of 13 test functions, but took almost all the best results in the Brown function (5D and 10D).

We have also compared the outcomes of SMGO with the competitor methods using the Wilcoxon and Kruskal-Wallis non-parametric statistical tests, pairing the results by the shared random initial point. Both test were performed with 5% significance level and with the null hypothesis that SMGO results are statistically similar to those from the respective competitor method. The two tests agreed that SMGO results were statistically similar to AdaLIPO for Styblinski-Tang (5D), Deb's #2 (5D, 10D), Schwefel (10D), and Brown (10D). Furthermore, similarities were found between SMGO and BayesianO for Deb's #2 (10D). Given these results, SMGO is found to have competitive results compared with the considered global optimization algorithms on the test functions employed in the analysis.

### *6.3. Computational time*

The computational time required to complete an iteration for each considered algorithms is analyzed for the case of the 5D Deb's function #1 optimization. Fig. 6 (in log scale) shows the time required to complete an iteration as $n$ increases. The mechanism of DIRECT calculates recursive hyperboxes to be sampled by batches, hence the computational time *per iteration* is not well defined, therefore results are not shown in the plot. For AdaLIPO an almost flat line is observed, with 10 to 100 $\mu$s per iteration, implying a computational time independent of the iteration number. SMGO has resulted in a polynomial complexity w.r.t. iterations, which is expected from the computational analysis presented in Section 5. Finally, BayesianO is the most demanding algorithm, requiring computational times per iteration between 2 and 3 orders of magnitude larger than SMGO in this test case.

The computational times observed for all the considered test functions have shown trends similar to those on the 5D Debs function #1. Table 4 summarizes the results, showing the total computational time required to complete an optimization run. The AdaLIPO algorithm, based on a simple mechanism of randomized generation of exploitation and exploration points, always shows very fast computational times, not affected by the dimensionality of the decision variable, scarcely affected by $n$, and taking 0.01-0.02 s per optimization run for all the test functions.

From the tests, BayesianO requires the largest computational times, which achieved around 4-5 s per iteration at $n = 500$ for most of the 10 dimensions functions. On the other hand, SMGO exhibits an intermediate per-iteration time around 100 $\mu$s to 10 ms for 5 dimensions functions and tens to hundreds of ms for 10 dimensions functions. Furthermore, for SMGO the total optimization run time when applied to 10 dimensions functions is around 10 times those employed for 5 dimensions functions. However, this is significantly faster than BayesianO, ranging from around 15 times shorter time for 10D functions, to 70 times shorter time for 5D functions.

The observed computational times, combined with the competitive optimization results presented

| **Test function** | $D$ | **DIRECT** | **AdaLIPO** | **SMGO** | **BayesianO** |
|---|---|---|---|---|---|
| Rosenbrock | 10 | 1.17 E +5 | 1.77 E +7 | **8.63 E +4** | 9.35 E +4 |
| Styblinski-Tang | 5 | -1.95 E +2 | -1.59 E +2 | -1.58 E +2 | **-1.95 E +2** |
| | 10 | -3.19 E +2 | -2.61 E +2 | -2.96 E +2 | **-3.33 E +2** |
| Deb's #1 | 5 | -5.29 E −1 | -8.29 E −1 | -8.07 E −1 | **-9.68 E −1** |
| | 10 | -2.11 E −1 | -6.74 E −1 | -6.97 E −1 | **-7.35 E −1** |
| Deb's #2 | 5 | -5.97 E −1 | -8.32 E −1 | -8.33 E −1 | **-8.59 E −1** |
| | 10 | -3.98 E −1 | -6.72 E −1 | **-6.81 E −1** | -6.77 E −1 |
| Schwefel | 5 | -1.47 E +3 | -1.28 E +3 | -1.23 E +3 | **-1.90 E +3** |
| | 10 | -1.48 E +3 | -1.83 E +3 | -1.79 E +3 | **-2.67 E +3** |
| Salomon | 5 | 3.45 E 0 | 2.64 E 0 | 2.19 E 0 | **6.38 E -1** |
| | 10 | 5.05 E 0 | 5.79 E 0 | 5.29 E 0 | **2.40 E 0** |
| Brown | 5 | **6.46 E -4** | 1.39 E -1 | 8.29 E -2 | 3.90 E -3 |
| | 10 | **4.76 E -2** | 9.78 E -1 | 9.61 E -1 | 1.65 E -1 |

Table 2: Results summary for comparative tests:
averages of $z^{*\langle n \rangle}$ over 100 runs, after 500 iterations per trial.

| **Test function** | $D$ | **DIRECT** | | **AdaLIPO** | | **SMGO** | | **BayesianO** | |
|---|---|---|---|---|---|---|---|---|---|
| | | 1st | 2nd | 1st | 2nd | 1st | 2nd | 1st | 2nd |
| Rosenbrock | 10 | 3 | 34 | 0 | 0 | **75** | 12 | 22 | **54** |
| Styblinski-Tang | 5 | 12 | **88** | 0 | 0 | 0 | 0 | **88** | 12 |
| | 10 | 15 | **71** | 0 | 0 | 8 | 15 | **77** | 14 |
| Deb's #1 | 5 | 0 | 0 | 2 | **61** | 2 | 35 | **96** | 4 |
| | 10 | 0 | 0 | 12 | **37** | 37 | 25 | **51** | 38 |
| Deb's #2 | 5 | 0 | 0 | 27 | 28 | 28 | **36** | **45** | 36 |
| | 10 | 0 | 0 | 29 | 28 | **44** | 25 | 27 | **47** |
| Schwefel | 5 | 0 | **82** | 1 | 8 | 0 | 9 | **99** | 1 |
| | 10 | 0 | 0 | 0 | **56** | 0 | 44 | **100** | 0 |
| Salomon | 5 | 0 | 0 | 0 | 30 | 0 | **70** | **100** | 0 |
| | 10 | 0 | **48** | 0 | 8 | 0 | 44 | **100** | 0 |
| Brown | 5 | **98** | 2 | 0 | 0 | 0 | 0 | 2 | **98** |
| | 10 | **100** | 0 | 0 | 0 | 0 | 0 | 0 | **100** |

Table 3: Results summary for comparative tests:
number of best and 2nd best trials for the tested algorithms

| **Test function** | $D$ | **AdaLIPO** | **SMGO** | **BayesianO** |
|---|---|---|---|---|
| Rosenbrock | 10 | 0.017 | 74.96 | 1071.04 |
| Styblinski-Tang | 5 | 0.013 | 7.71 | 511.66 |
| | 10 | 0.014 | 75.01 | 1085.11 |
| Deb's #1 | 5 | 0.012 | 7.71 | 574.08 |
| | 10 | 0.014 | 74.64 | 1164.80 |
| Deb's #2 | 5 | 0.012 | 7.72 | 576.63 |
| | 10 | 0.014 | 73.97 | 1176.10 |
| Schwefel | 5 | 0.011 | 7.77 | 583.24 |
| | 10 | 0.014 | 75.13 | 1188.08 |
| Salomon | 5 | 0.013 | 7.63 | 643.25 |
| | 10 | 0.015 | 75.08 | 1033.54 |
| Brown | 5 | 0.012 | 7.82 | 476.40 |
| | 10 | 0.013 | 75.47 | 886.81 |

Table 4: Comparison of total optimization run times, averaged over the
100 independent runs for each test case (all in seconds)

in Figs. 4-5, and Table 2, indicate that the proposed SMGO algorithm shows a good trade-off between optimization performance and computational speed, compared to the other considered methods.

## 7. Conclusions and Further Work

In this work a sequential algorithm for global optimization of black-box functions has been proposed. The SMGO algorithm assumes a Lipschitz-

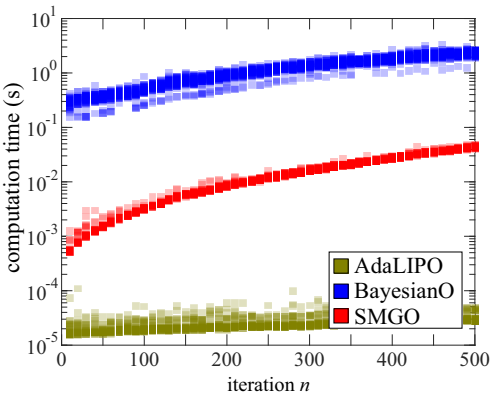


Figure 6: Calculation times on 5D Deb's #1 (log scale)

continuous cost function and is based on a nonlinear Set Membership function approximation. The selection of the test points where the black-box function is evaluated is carried out by solving simplified geometry-based problems on the guaranteed lower bounds or uncertainty intervals of the function, provided by a Set Membership model. It is shown that the SMGO algorithm finds a suboptimal solution with desired tolerance after finite iterations and converges asymptotically to the global minimum. Furthermore, the computational complexity of the algorithm is discussed and a cache-based solution is introduced to improve the computational burden.

The proposed method is evaluated against other representative global optimization methods on several test functions, comparing the quality of the best solutions found after a fixed number of calls to the black-box function. The results show the competitiveness of the SMGO algorithm compared to state of the art methods found in literature. Ongoing research aims to extend the algorithm to include constraints and its evaluation on experimental applications.

## References


[1] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient Global Optimization of Expensive Black-Box Functions," *Journal of Global Optimization*, vol. 13, pp. 455–492, 1998.

[2] N. Pham, A. Malinowski, and T. Bartczak, "Comparative study of derivative free optimization algorithms," *IEEE Transactions on Industrial Informatics*, vol. 7, no. 4, pp. 592–600, 2011.

[3] S. Amaran, N. V. Sahinidis, B. Sharda, and S. J. Bury, "Simulation optimization: a review of algorithms and applications," *Annals of Operations Research*, vol. 240, no. 1, pp. 351–380, 2016.

[4] S. Gao, L. Shi, and Z. Zhang, "A peak-over-threshold search method for global optimization," *Automatica*, vol. 89, pp. 83 – 91, 2018.

[5] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4. IEEE, 2014, pp. 1942–1948.

[6] Q. Liu, S. Yang, and J. Wang, "A collective neurodynamic approach to distributed constrained optimization," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 8, pp. 1747–1758, 2017.

[7] H. Han, W. Lu, and J. Qiao, "An Adaptive Multiobjective Particle Swarm Optimization Based on Multiple Adaptive Methods," *IEEE Transactions on Cybernetics*, vol. 47, no. 9, pp. 2754–2767, 2017.

[8] M. J. Powell, "UOBYQA: Unconstrained optimization by quadratic approximation," *Mathematical Programming, Series B*, vol. 92, no. 3, pp. 555–582, 2002.

[9] A. Bemporad, "Global optimization via inverse distance weighting and radial basis functions," *Computational Optimization and Applications*, vol. 77, no. 2, pp. 571–595, 2020.

[10] Y. Wang, D. Q. Yin, S. Yang, and G. Sun, "Global and local surrogate-assisted differential evolution for expensive constrained optimization problems with inequality constraints," *IEEE Transactions on Cybernetics*, vol. 49, no. 5, pp. 1642–1656, 2019.

[11] S. Piyavskii, "An algorithm for finding the absolute extremum of a function," *USSR Computational Mathematics and Mathematical Physics*, vol. 12, no. 4, pp. 57 – 67, 1972.

[12] B. O. Shubert, "A sequential method seeking the global maximum of a function," *SIAM Journal on Numerical Analysis*, vol. 9, no. 3, pp. 379–388, 1972.

[13] D. R. Jones, C. D. Perttunen, and B. E. Stuckman, "Lipschitzian optimization without the Lipschitz constant," *Journal of Optimization Theory and Applications*, vol. 79, no. 1, pp. 157–181, oct 1993.

[14] D. E. Finkel and C. T. Kelley, "Additive Scaling and the DIRECT Algorithm," *Journal of Global Optimization*, vol. 36, no. 4, pp. 597–608, oct 2006.

[15] S. Z. Khong, D. Nešić, C. Manzie, and Y. Tan, "Multidimensional global extremum seeking via the direct optimisation algorithm," *Automatica*, vol. 49, no. 7, pp. 1970 – 1978, 2013.

[16] R. Paulavičius and J. Žilinskas, "Simplicial Lipschitz optimization without the Lipschitz constant," *Journal of Global Optimization*, vol. 59, no. 1, pp. 23–40, may 2014.

[17] R. Paulavičius, Y. D. Sergeyev, D. E. Kvasov, and J. Žilinskas, "Globally-biased DISIMPL algorithm for expensive global optimization," pp. 545–567, 2014.

[18] C. Malherbe and N. Vayatis, "Global optimization of Lipschitz functions," *34th International Conference on Machine Learning, ICML 2017*, vol. 5, pp. 3592–3601, 2017.

[19] M. Milanese and C. Novara, "Unified set membership theory for identification, prediction and filtering of nonlinear systems," *Automatica*, vol. 47, no. 10, pp. 2141 – 2151, 2011.

[20] ——, "Set Membership identification of nonlinear systems," *Automatica*, vol. 40, no. 6, pp. 957–975, 2004.

[21] C. Novara, F. Ruiz, and M. Milanese, "Direct filtering: A new approach to optimal filter design for nonlinear systems," *IEEE Transactions on Automatic Control*,

vol. 58, no. 1, pp. 86–99, 2013.

[22] L. Fagiano and C. Novara, "Learning a Nonlinear Controller From Data: Theory, Computation, and Experimental Results," *IEEE Transactions on Automatic Control*, vol. 61, no. 7, pp. 1854–1868, jul 2016.

[23] L. Sabug, F. Ruiz, and L. Fagiano, "On the use of Set Membership theory for global optimization of black-box functions," in *Proceedings of the 59$^{th}$ Conf. Decision and Control (CDC)*. IEEE, 2020.

[24] M. Milanese and C. Novara, "Computation of local radius of information in SM-IBC identification of nonlinear systems," *Journal of Complexity*, vol. 23, no. 4-6, pp. 937–951, 2007.

[25] V. Beiranvand, W. Hare, and Y. Lucet, "Best practices for comparing optimization algorithms," *Optimization and Engineering*, vol. 18, no. 4, pp. 815–848, sep 2017.

[26] Y.-W. Shang and Y.-H. Qiu, "A Note on the Extended Rosenbrock Function," *Evolutionary Computation*, vol. 14, no. 1, pp. 119–126, mar 2006.