Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 56-2 (2023) 3966–3971

A Set Membership approach to black-box optimization for time-varying problems ⋆

Lorenzo Sabug Jr., Fredy Ruiz, Lorenzo Fagiano

Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Piazza Leonardo da Vinci 32, Milano, Italy (e-mail: {lorenzojr.sabug, fredy.ruiz, lorenzo.fagiano}@polimi.it).

Abstract. A novel method to tackle black-box optimization for time-varying problems is proposed. Using a Set Membership (SM) framework, the approach directly adjusts the uncertainty associated with old data points as new samples are introduced. Uninformative old samples are discarded, and the adjusted model guides the exploitation and exploration routines as characteristic of black-box optimization. With the proposed method, there is no need to estimate the time-related rate of change of the hidden function, as required in previous literature. We provide results of a benchmark test, comparing the performance of the proposed method to other approaches to time-varying black-box optimization, with promising results.

Keywords: time-varying systems, modeling and identification, non-parametric methods, non-convex optimization, set membership, data-driven optimization

1. INTRODUCTION

In many industrial fields and scientific/technological domains, we are faced with non-convex optimization problems where the objective and/or constraints are black-box functions (BBF). They are referred as "black-box" due to the lack of closed-form expressions mapping the input variables to the output. Moreover, objective and constraint models whose complexity is prohibitive to be handled analytically are also commonly treated as BBF. Such functions arise in many fields, such as automotive Silvas et al. (2016), aerospace Blasi et al. (2013), and agricultural processes Picheny et al. (2017), among many others.

There is a rich literature for black-box optimization (BBO) in the time-invariant case, which entails the search for an optimal value and point for a black-box function. Different problem settings have been considered, whether in the unconstrained case Jones et al. (1998); Finkel and Kelley (2006); Malherbe and Vayatis (2017); Sabug et al. (2021), with explicit constraints Bemporad (2020), and even for constraints which are black-box Gelbart et al. (2014); Antonio (2019); Hernández-Lobato et al. (2016); Sabug et al. (2022). However, there are processes in which the underlying phenomena changes over time, in a setting we refer to as "time-varying" (TV), caused by plant degradation, prolonged use, or interactions with the environment. In this case, the corresponding BBF changes over time, leading to change in optimal value, and/or movement of the optimal point within the search space. There have been works which deal with TV-BBO for aerospace design, evolutionary robotics, and path planning, among others, as documented in the survey Cruz et al. (2011). Most initial approaches are based on swarm-based Blackwell (2007); Fernandes et al. (2008) and evolutionary techniques Barrico and Antunes (2007); Bosman (2007), among related methods, which assume that multiple function evaluations can be taken in parallel at each iteration, thus decoupling the BBF time variations from the function evaluations. However, in most real cases, a sequential sampling over time is mandatory, so that each sampled value pertains potentially to a different function, due to its variation over time.

As the interest for Gaussian process (GP)-based methods and Bayesian optimization (BO) rose in the past decade, there have been several efforts to extend BO for solving TV-BBO problems. The TV-related extensions propose approaches to address the *forgetting-remembering trade-off* of data samples based on their informativeness, modifying the GP model to approximate the actual time-varying function. The updated model is then used to address the existing trade-off between exploitation and exploration, which is shared with time-invariant BBO. Bogunovic et al. (2016) considered two approaches to the forgetting-remembering trade-off. The first is using a resetting mechanism, running BO by batch and clearing the data set used in the GP modelling when a batch is complete. This means that only the samples acquired in the latest batch are considered in the construction of the GP model. As noted in the same work Bogunovic et al. (2016), the batch-based update of model only wastes the data set from the previous batches, while the underlying function is changing smoothly in most cases. It then proposed the Time-Varying GP Upper Confidence Bound (TV-GP-UCB) as the second approach, implementing a smooth forgetting mechanism by using a GP posterior update model. This model requires/assumes a hyperparameter ϵ_{TV}, described as a forgetting factor. Their proposed TV-GP-UCB presents a limitation because ϵ_{TV} needs to be assumed, and the process evolution might not always match with the GP posterior update model with this factor. In Zhou and Shroff (2021), in addition to the reset-based GP-based optimization, it also considered the use of a fixed sliding window, which smoothly forgets older samples as new ones are introduced. While the proposed mechanism is intuitive and simple to implement, forgetting

⋆ The first author gratefully acknowledges the support of Department of Science and Technology–Science Education Institute (DOST-SEI) of the Philippines for his research.
This research has been partially supported by the Italian Ministry of University and Research (MIUR) under the PRIN 2017 grant n. 201732RS94 "Systems of Tethered Multicopters", and by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 953348 "ELO-X - Embedded Learning and Optimization for the neXt generation of smart industrial control systems".

10.1016/j.ifacol.2023.10.1343

of samples due to age ignores the informativeness that these samples can still introduce even when old, e.g. in the static areas of the search space. There were also recent BO-based extensions for problem setups where both the sampling point and timing are to be optimized Nyikosa et al. (2018), and where the evaluation times are not constant Imamura et al. (2020); however, these problem contexts are out of the scope of this paper.

We propose in this paper a novel black-box optimization method for time-varying problems. To deal with the remembering-forgetting trade-off, we build on recent results Sabug et al. (2021, 2022) and use the Set Membership framework to directly assess the informativeness of old samples, given the incoming data. The updates, in turn, are used to discard uninformative old samples when necessary. This differs from existing methods in that it does not need/assume a time-based rate-of-change (which is in most cases not available). For clarity of presentation, the proposed method is discussed in the context of unconstrained optimization problems. Note, however, that the proposed time-varying model update mechanism is transferable to time-varying constraint functions with relatively minor modifications.

This paper is organized as follows: Section 2 states the problem settings and assumptions used in this work. Section 3 introduces the Set Membership Global Optimization method from Sabug et al. (2022). Section 4 describes the direct data-driven model updates for time-varying black-box functions. Section 5 compares the performance of the proposed time-varying optimization method with the ones from literature, and we conclude in Section 6.

2. PROBLEM STATEMENT

We consider a scalar function $f(\boldsymbol{x}, \tau)$, dependent on the sampling location $\boldsymbol{x} \in \mathcal{X}$, as well as time $\tau \in \mathbb{R}_+$. The search space $\mathcal{X}$ is a compact convex set in $\mathbb{R}^D$ with D being the dimensionality of the problem. f has no analytical expression, and the only *a priori* knowledge about f is given by the following assumptions:

Assumption 1. *At any $\tau \in \mathbb{R}_+$, f is Lipschitz-continuous with unknown but finite constant γ, i.e.*

$$|f(\boldsymbol{x}_1, \tau) - f(\boldsymbol{x}_2, \tau)| \leq \gamma \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|, \ \forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{X}.$$

Assumption 2. *At any $\boldsymbol{x} \in \mathcal{X}$, f evolves with a finite rate of change γ_τ:*

$$|f(\boldsymbol{x}, \tau_1) - f(\boldsymbol{x}, \tau_2)| \leq \gamma_\tau |\tau_1 - \tau_2|, \ \forall \tau_1, \tau_2 \in \mathbb{R}_+.$$

Assumption 2 simply states that f does not jump from one form to another, i.e., f changes gradually with τ. Lastly, we assume that we can acquire individual values of f by sampling:

Assumption 3. *Given $\boldsymbol{x} \in \mathcal{X}$ and $\tau \in \mathbb{R}_+$, f can be sampled with finite and time-invariant noise bound $\bar{\epsilon}$:*

$$z = f(\boldsymbol{x}, \tau) + \epsilon, |\epsilon| \leq \bar{\epsilon} < \infty.$$

Now we state the problem addressed in this paper.

Problem 1. *Design an algorithm that generates a sequence of points $\{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \dots\}, \boldsymbol{x}^{(i)} \in \mathcal{X}$, to search and track a time-varying minimizer point $\boldsymbol{x}^*(\tau)$ of f, such that*

$$\boldsymbol{x}^*(\tau) = \arg\min_{\boldsymbol{x} \in \mathcal{X}} f(\boldsymbol{x}, \tau).$$

3. SET MEMBERSHIP GLOBAL OPTIMIZATION (SMGO-Δ)

We give an overview of the considered optimization algorithm SMGO-Δ from Sabug et al. (2022), for which we propose the model update mechanism for time-varying problems. For simplicity of discussion, we introduce the method in the context of unconstrained and time-invariant problems.

3.1 Set Membership (SM) model from data

At each iteration $n = 1, \dots, N$, a point $\boldsymbol{x}^{(n)} \in \mathcal{X}$ is sampled, either via simulation, experiment, or a combination of both. Such an evaluation results in a tuple $\breve{\boldsymbol{x}}^{(n)} := \left(\boldsymbol{x}^{(n)}, z^{(n)}\right)$. The resulting tuple is added to the existing data points to iteratively construct the unordered data set $\boldsymbol{X}^{\langle n \rangle}$:

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

$\boldsymbol{X}^{\langle n \rangle}$ only contains the *valid* tuples up to the latest sample index n, because some might have been discarded/forgotten due to the mechanism to be discussed in the next section. As a result, the cardinality of $\boldsymbol{X}^{\langle n \rangle}$ is not necessarily equal to n.

Given a known Lipschitz constant γ and noise bound $\bar{\epsilon}$, we can build the SM-based upper- and lower-bound functions for the underlying objective f following Milanese and Novara (2004):

$$\overline{f}^{\langle n \rangle}(\boldsymbol{x}) = \min_{\breve{\boldsymbol{x}}^{(k)} \in \boldsymbol{X}^{\langle n \rangle}} \left(z^{(k)} + \epsilon + \gamma \|\boldsymbol{x} - \boldsymbol{x}^{(k)}\|\right), \tag{1}$$

$$\underline{f}^{\langle n \rangle}(\boldsymbol{x}) = \max_{\breve{\boldsymbol{x}}^{(k)} \in \boldsymbol{X}^{\langle n \rangle}} \left(z^{(k)} - \epsilon - \gamma \|\boldsymbol{x} - \boldsymbol{x}^{(k)}\|\right). \tag{2}$$

Furthermore, we can obtain a central approximation of f,

$$\tilde{f}^{\langle n \rangle}(\boldsymbol{x}) = \frac{1}{2}\left(\overline{f}^{\langle n \rangle}(\boldsymbol{x}) + \underline{f}^{\langle n \rangle}(\boldsymbol{x})\right), \tag{3}$$

and its corresponding uncertainty measure

$$\lambda^{\langle n \rangle}(\boldsymbol{x}) = \overline{f}^{\langle n \rangle}(\boldsymbol{x}) - \underline{f}^{\langle n \rangle}(\boldsymbol{x}). \tag{4}$$

Since in practice the quantities γ and noise bound $\bar{\epsilon}$ are unknown, the optimization algorithm computes and updates their estimates from data as well, see Sabug et al. (2022) for details.

3.2 Generation of candidate points

From the sampled locations $\boldsymbol{x}^{(i)} \in \boldsymbol{X}^{\langle n \rangle}$, we methodically generate a set of candidate points $\boldsymbol{E}^{\langle n \rangle}$ around $\mathcal{X}$, from which we choose the next sampling point by running the exploitation and (if necessary) exploration routines. Details regarding the candidate points generation can be found in Sabug et al. (2022).

3.3 Exploitation

We now consider the best tuple from $\boldsymbol{X}^{\langle n \rangle}$, defined as

$$\breve{\boldsymbol{x}}^{*\langle n \rangle} = \left(\boldsymbol{x}^{*\langle n \rangle}, z^{*\langle n \rangle}\right) := \arg\min_{\breve{\boldsymbol{x}}^{(i)} \in \boldsymbol{X}^{\langle n \rangle}} z^{(i)}. \tag{5}$$

Given $\boldsymbol{x}^{*\langle n \rangle}$, we attempt to find a candidate point with maximum predicted improvement w.r.t. the current best one $z^{*\langle n \rangle}$. For this, we perform a optimization on a small trust region $\mathcal{T}^{\langle n \rangle}$ around $\boldsymbol{x}^{*\langle n \rangle}$, and using a combination of the SM-based center approximation and uncertainty,

$$\boldsymbol{x}_\theta^{\langle n \rangle} = \arg\min_{\boldsymbol{x} \in \boldsymbol{E}^{\langle n \rangle} \cap \mathcal{T}^{\langle n \rangle}} \tilde{f}^{\langle n \rangle}(\boldsymbol{x}) - \beta \lambda^{\langle n \rangle}(\boldsymbol{x}) \tag{6}$$

with a weighting parameter β. The selected $\boldsymbol{x}_\theta^{\langle n \rangle}$ is then subjected to an *expected improvement test* to evaluate worthiness for evaluation

$$\underline{f}^{\langle n \rangle}(\boldsymbol{x}_\theta^{\langle n \rangle}) \leq z^{*\langle n \rangle} - \eta, \tag{7}$$

with $\eta = \alpha\tilde{\gamma}^{\langle n\rangle}$ being the *minimum improvement threshold*, and a parameter $\alpha > 0$. If (7) is satisfied, $\boldsymbol{x}_\theta^{\langle n\rangle}$ is assigned as the next evaluation point $\boldsymbol{x}^{(n+1)}$. Otherwise, SMGO-Δ proceeds to the exploration routine.

3.4 Exploration

The rationale of this routine is to discover the shape of the $f(\boldsymbol{x})$, whose added information can be used for succeeding iterations. We select the exploration point $\boldsymbol{x}_\phi^{\langle n\rangle}$,

$$\boldsymbol{x}_\phi^{\langle n\rangle} = \arg\max_{\boldsymbol{x}\in\boldsymbol{E}^{\langle n\rangle}} \xi_\phi(\boldsymbol{x}), \tag{8}$$

where $\xi_\phi(\boldsymbol{x})$ is the exploration merit function

$$\xi_\phi(\boldsymbol{x}) \coloneqq d(\boldsymbol{x})\lambda^{\langle n\rangle}(\boldsymbol{x}) + k\nu^{\langle n\rangle}(\boldsymbol{x}). \tag{9}$$

The first expression in (9) is composed of the uncertainty $\lambda^{\langle n\rangle}(\boldsymbol{x})$ (see (4)) and remoteness

$$d(\boldsymbol{x}) \coloneqq \min_{\boldsymbol{x}^{(i)}\in\boldsymbol{X}^{\langle n\rangle}} \left\|\boldsymbol{x} - \boldsymbol{x}^{(i)}\right\|.$$

Meanwhile, the second expression in (9) is a small factor $k = 1\times10^{-6}$ multiplied with the candidate point age $\nu^{\langle n\rangle}(\boldsymbol{x})$, ensuring theoretical convergence of the algorithm for time-invariant problems (as proven in Sabug et al. (2022)). The point selected in this routine is directly selected as the next sampling point $\boldsymbol{x}^{(n+1)}$.

4. SM-BASED DIRECT MODEL UPDATES FROM DATA

The common challenge in optimizing time-varying problems is to detect time-related changes to f. In the SM-based framework, this translates to recognizing whether:

- the variation of the incoming data serves to refine the model of f w.r.t. $\boldsymbol{x}$, or
- if this variation is instead due to the evolution of $f(\boldsymbol{x})$ w.r.t. τ.

In the proposed method, given an initial model, we attempt to recognize whether incoming samples are due to a time evolution of f. This is achieved directly from data and the SM-based bounds, without need to estimate the time-related rate-of-change of the function as required in Bogunovic et al. (2016).

4.1 Initial SM-Based Model Training

In the proposed method, we are assumed to be given an initial data set $\boldsymbol{X}^{\langle 0\rangle}$ to build our SM model, which may be from an initial model characterization. $\boldsymbol{X}^{\langle 0\rangle}$ can be acquired using a user-preferred sampling method, e.g., space-filling methods, pseudo-random sampling, or by running an optimization method (e.g., SMGO-Δ for time-invariant problems), which has the added advantage of simultaneously identifying a best point in $\boldsymbol{x}^{*\langle 0\rangle}$.

This initial training serves mainly to compute the noise bound (resp. Lipschitz constant) estimate $\tilde{\epsilon}$ (resp. $\tilde{\gamma}^{\langle 0\rangle}$), if their true values are not available *a priori*. While the noise bound estimate $\tilde{\epsilon}$ is held constant throughout the optimization process, the Lipschitz constant estimate $\tilde{\gamma}^{\langle n\rangle}$ can be adjusted as necessary, as will be discussed in Section 4.4.

4.2 SM Model Adjustment

We adjust the SM-based model using the intuition that if a new data point violates the existing SM-based model, the function is deemed changing w.r.t. τ. In addition, in updating the SM-based bounds $\overline{f}^{\langle n\rangle}$ and $\underline{f}^{\langle n\rangle}$, we should trust the new data more than the older ones. From the difference in the bounds from the existing model and the new sample, we reflect the increasing distrust of the old data by increasing their respective uncertainties.

Consider an existing data set $\boldsymbol{X}^{\langle n-1\rangle}$, composed of data collected before present time $\tau = t$, and from which we have built our SM-based models characterized by $\overline{f}^{\langle n-1\rangle}(\boldsymbol{x})$ and $\underline{f}^{\langle n-1\rangle}(\boldsymbol{x})$. Suppose that we have an incoming data point $\mathring{\boldsymbol{x}}^{(n)} \coloneqq \left(\boldsymbol{x}^{(n)}, z^{(n)}, \tau^{(n)} = t\right)$ from evaluation, which is added to $\boldsymbol{X}^{\langle n-1\rangle}$. Using $\tilde{\epsilon}$ and $\tilde{\gamma}^{\langle n\rangle}$, we consider the SM-based bounds caused *solely* by $\mathring{\boldsymbol{x}}^{(n)}$,

$$\overline{f}^{(n)}(\boldsymbol{x}) = z^{(n)} + \tilde{\epsilon} + \tilde{\gamma}^{\langle n\rangle}\left\|\boldsymbol{x} - \boldsymbol{x}^{(n)}\right\|, \tag{10}$$

$$\underline{f}^{(n)}(\boldsymbol{x}) = z^{(n)} - \tilde{\epsilon} - \tilde{\gamma}^{\langle n\rangle}\left\|\boldsymbol{x} - \boldsymbol{x}^{(n)}\right\|. \tag{11}$$

By intuition, we want to trust the new sample $\mathring{\boldsymbol{x}}^{(n)}$, because it is the most recent one for the point $\boldsymbol{x}^{(n)}$. Hence, we can assert that the SM-based bounds at $\boldsymbol{x}^{(n)}$ are actually $z^{(n)} + \tilde{\epsilon}$ (upper) and $z^{(n)} - \tilde{\epsilon}$ (lower). As a consequence, we adjust the existing bounds such that this assertion is compatible to the updated model. Figure 1 shows a case when the new sample fits inside the previous bounds, i.e.,

$$\forall\boldsymbol{x}\in\mathcal{X},\ \overline{f}^{(n)}(\boldsymbol{x}) \le \overline{f}^{\langle n-1\rangle}(\boldsymbol{x}) \tag{12}$$

$$\underline{f}^{(n)}(\boldsymbol{x}) \ge \underline{f}^{\langle n-1\rangle}(\boldsymbol{x}). \tag{13}$$

In such case, we interpret that we are only refining the existing model using $\mathring{\boldsymbol{x}}^{(n)}$. However, if $\overline{f}^{(n)}(\boldsymbol{x}) \ge \overline{f}^{\langle n-1\rangle}(\boldsymbol{x})$ and/or $\underline{f}^{(n)}(\boldsymbol{x}) \le \underline{f}^{\langle n-1\rangle}(\boldsymbol{x})$ [Figure 1], we adjust the existing bounds $\overline{f}^{\langle n-1\rangle}(\boldsymbol{x})$, $\underline{f}^{\langle n-1\rangle}(\boldsymbol{x})$ such that

$$\overline{f}^{'\langle n-1\rangle}(\boldsymbol{x}) = \overline{f}^{\langle n-1\rangle}(\boldsymbol{x}) + \lambda^{'\langle n\rangle} \tag{14}$$

$$\underline{f}^{'\langle n-1\rangle}(\boldsymbol{x}) = \underline{f}^{\langle n-1\rangle}(\boldsymbol{x}) - \lambda^{'\langle n\rangle} \tag{15}$$

where

$$\lambda^{'\langle n\rangle} = \max\left(\begin{bmatrix} 0 \\ \left(z^{(n)} + \tilde{\epsilon}\right) - \overline{f}^{\langle n-1\rangle}\left(\boldsymbol{x}^{(n)}\right) \\ \underline{f}^{\langle n-1\rangle}\left(\boldsymbol{x}^{(n)}\right) - \left(z^{(n)} - \tilde{\epsilon}\right) \end{bmatrix}\right). \tag{16}$$

The physical significance of this is that we increase the uncertainty ("distrust") of the old data if the SM bound (upper or lower) from the newly-introduced $\mathring{\boldsymbol{x}}^{(n)}$ exceeds that of the existing model. Consequently, this means that every new data point that exceeds the existing bounds decrease the informativeness of the old ones, motivating the mechanism for forgetting/discarding data points.

4.3 Discarding Uninformative Samples

We propose a new method for discarding samples from the data set $\boldsymbol{X}^{\langle n\rangle}$. In summary, we discard a sample $\mathring{\boldsymbol{x}}^{(j)}$ that is not informative enough to decide any of the SM-based bounds anywhere in the search space, i.e., it is a *redundant* sample. In this method, we forget samples $\mathring{\boldsymbol{x}}^{(j)}$ meeting one of the following criteria:

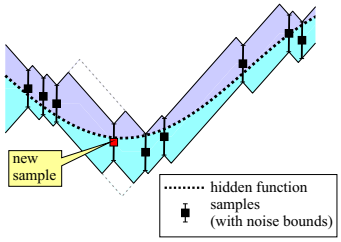

Figure 1. Refining an existing model using a new sample.

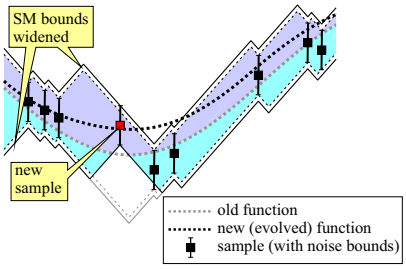

Figure 2. Adjustment of SM bounds uncertainty with a new data point.

(1) older than a minimum threshold, $t - \tau^{(j)} \geq \overline{T}_y$ (i.e. "old enough"), and whose own upper- *and* lower bounds are not anymore the tightest at location $\boldsymbol{x}^{(j)}$;

(2) older than a maximum threshold age, $t - \tau^{(j)} \geq \overline{T}_g$ (i.e. "too old"),

such that $\overline{T}_g > \overline{T}_y$. The first criterion for forgetting samples is new in this work, which uses SM-based bounds to quantify the informativeness of the samples. The second criterion, on the other hand, is similar to the "sliding time window" approach, as referred in Zhou and Shroff (2021). Given the above two criteria, we actually have two sliding windows, the one which makes old samples eligible for discarding, and another which force-discards very old samples.

4.4 Implementation aspects

Updates for Lipschitz constant estimate. To treat the case of hidden functions whose evolution involves a changing Lipschitz constant, e.g. a bump evolving out of a flat function, we propose a workaround to make sure that the estimate $\tilde{\gamma}^{\langle n \rangle}$ follows that of the hidden function.

Consider a developing bump in f, increasing its true Lipschitz constant γ. As new samples are introduced from this developing bump, the SM-based uncertainty of the old model continues to widen by virtue of (14)-(15). To track the increase of the Lipschitz constant in this case, we propose the use of a variable $\breve{\lambda}$, which accumulates the uncertainty increments $\lambda^{\langle n \rangle}$ added to the upper- and lower bounds (see (16)). The updates will be as follows

$$\breve{\lambda}^{\langle n+1 \rangle} = \breve{\lambda}^{\langle n \rangle} + \lambda'^{\langle n \rangle}.$$

Once $\breve{\lambda}^{\langle n \rangle}$ reaches a threshold $\bar{\lambda}$ (which we set here as $\bar{\lambda} = 10\tilde{\epsilon}$), we recalculate the Lipschitz constant estimate $\tilde{\gamma}^{\langle n \rangle}$ according to the existing valid data.

Candidate points filtering. The proposed candidate points generation scheme for SMGO-Δ, as pointed out in Sabug et al. (2022), results in candidate points quantity of $\mathcal{O}(Dn + n^2)$ w.r.t. dimensionality D and iterations n, introducing a bottleneck for calculating (6) and (8) mainly due to increasing n. To minimize the computational burden, we propose a heuristic to filter out candidate points to limit their quantity. Minimum computational times are especially important in cases with endless streaming samples, as anticipated in time-varying optimization contexts.

The main argument behind the proposed approach is we can ignore or discard candidate points which are not likely to be sampled by exploration (8) within some given iteration horizon. To proceed, we analyze the terms in the exploration merit function (9).

- the remoteness $d(\boldsymbol{e})$ of the candidate points w.r.t. existing samples, is mostly non-increasing: assuming that samples are never forgotten once introduced, a candidate point will get nearer to a sample. However, in the time-varying case where $\overline{T}_y$ and $\overline{T}_g$ are large, this can be a reasonable assumption as well.
- the uncertainty measure $w_\lambda(\boldsymbol{e})$ is also mostly non-increasing: with many samples, it can be claimed that $\tilde{\gamma}^{\langle n \rangle}$ has stabilized. Hence, its major change mechanism is triggered when $\lambda^{\langle n \rangle}(\boldsymbol{x})$ decreases (due to new nearby samples).

Given these information, we can infer if a certain candidate point is worthy of keeping in the memory, or if we should filter it out. Consider an iteration horizon of $N_{\boldsymbol{M}} > 0$, which can be equal to the total iteration budget N, or any fairly large integer. Furthermore, consider an implied ranking of candidate points in $\boldsymbol{E}^{\langle n \rangle}$, in order of decreasing exploration merit

$$\boldsymbol{M}^{\langle n \rangle} \coloneqq \left\{ \boldsymbol{m}_1^{\langle n \rangle}, \boldsymbol{m}_2^{\langle n \rangle}, \boldsymbol{m}_3^{\langle n \rangle}, \dots \right\}. \tag{17}$$

Given the ranking $\boldsymbol{M}^{\langle n \rangle}$, we propose to filter out a candidate point $\boldsymbol{e}$ if

$$\bar{\xi}^{\langle n+N_{\boldsymbol{M}} \rangle}(\boldsymbol{e}) \coloneqq d^{\langle n \rangle}(\boldsymbol{e}) \lambda^{\langle n \rangle}(\boldsymbol{e}) + k\tau^{\langle n+N_{\boldsymbol{M}} \rangle}(\boldsymbol{e}) \tag{18}$$

$$< \xi_\phi^{\langle n \rangle} \left(\boldsymbol{m}_{N_{\boldsymbol{M}}}^{\langle n \rangle} \right). \tag{19}$$

The above condition means that if $\boldsymbol{e}$, after $N_{\boldsymbol{M}}$ iterations, has a best-case predicted merit $\bar{\xi}^{\langle n+N_{\boldsymbol{M}} \rangle}(\boldsymbol{e})$ that is still not placed in the first $N_{\boldsymbol{M}}$ entries of $\boldsymbol{M}^{\langle n \rangle}$, we delete $\boldsymbol{e}$ from $\boldsymbol{E}^{\langle n \rangle}$.

We note that filtering action should only be performed when the number of samples is large enough. If the filtering is started at low n, the merit of the first-ranked candidate point will be very large relative to the other points due to high uncertainty (as expected with low number of samples), and newly-generated candidate points will not be retained. In this paper, we set a horizon value of $N_{\boldsymbol{M}} = 500$, which is also the minimum n to enable candidate points filtering.

5. SYNTHETIC BENCHMARK TESTS

5.1 Generation of benchmark problems

We consider different problems in this test, all of which are based on the moving peaks benchmark (MPB) problem Yazdani et al. (2022); Nyikosa et al. (2018), the most commonly-used problem in the time-varying optimization literature. The basic form that we use in the tests is

$$f_a(\boldsymbol{x}, \tau) = - \max_{i \in \{1,\dots,m\}} \frac{h_i(\tau)}{1 + w_i(\tau) \left(\boldsymbol{x}_i - \boldsymbol{c}_i(\tau)\right)^\top \boldsymbol{W}_i(\tau) \left(\boldsymbol{x}_i - \boldsymbol{c}_i(\tau)\right)}, \tag{20}$$

where for each peak i, $h_i(\tau)$ is the peak height, $\boldsymbol{c}_i(\tau)$ is the center, $w_i(\tau)$ is the weight, and $\boldsymbol{W}_i(\tau)$ is a diagonal matrix implementing the condition number κ_i of the peak. We have considered testbench problems with $D = 3$ and $D = 5$, with each problem displaying 6 and 10 moving peaks, respectively, and a search space of $\mathcal{X} = [-10, 10]^D$. We have considered several characteristics, such as applying different randomly-generated time-varying heights $h_i(\tau)$, higher condition numbers κ_i (ill-conditioning), and time-varying $\kappa_i(\tau)$ to each peak i. The properties of the tested functions are tabulated in Table 1.

Problem	D	**Time-varying** $h_i(\tau)$	$\kappa_i > 1$	**Time-varying** κ_i
P1	3	✓		
P2	3	✓	✓	
P3	3	✓	✓	✓
P4	5	✓		
P5	5	✓	✓	
P6	5	✓	✓	✓

Table 1. Objective function properties for the time-varying optimization benchmark tests

In contrast to the benchmarks described in Yazdani et al. (2022), we do not send any change signal to the optimization algorithms. This means that the compared methods should automatically update their models according to the (gradual) change in the hidden functions.

5.2 Optimization tests

To solve the given problems, we consider the following approaches for time-varying optimization:

- BO, with batch-based reset (BO-R) Zhou and Shroff (2021); Bogunovic et al. (2016)
- BO, with sliding window (BO-W) Zhou and Shroff (2021)
- SMGO-Δ, with time-varying updates (this work)

We assume that all samplings are sequential, i.e. only one sampling is allowed at every time step, and we can use the time step counter τ to also mean the iteration number n. Furthermore, we set the batch length for BO-R as $125D$, which we also set as the sliding window length for BO-W. On the other hand, we set SMGO time-varying hyperparameters as $\overline{T}_y = 125D$ and $\overline{T}_g = 2\overline{T}_y$. Furthermore, we set SMGO-Δ-specific parameters $\beta = 0.1$ and $\alpha = 0.005$, following Sabug et al. (2022).

For all the compared methods, are given an initial budget of $125D$ evaluations to build an initial model and optimize the initial objective function. Afterwards, we consider a frame of $\overline{T}_{tv} = 1000D$ time steps, during which we allow the objectives to gradually evolve while we perform the time-varying optimization. To enable statistical comparison, we performed 25 independent optimization runs for each problem and method, using the same set of starting points to maintain fairness of comparison.

5.3 Comparative results: optimization performance

To compare performance of the optimization algorithms, we first consider the optimality gap, which is the difference between the latest identified best value and the actual function minimizer at time step τ:

$$\delta(\tau) \coloneqq f(\boldsymbol{x}^{*\langle\tau\rangle}, \tau) - f(\boldsymbol{x}^*(\tau), \tau). \tag{21}$$

In this test, we consider the optimality gap throughout each run, and collect information on all runs. Now, consider a run r on a test problem. The average optimality gap over a run r is given by $\tilde{\delta}_r = \frac{1}{\overline{T}_{tv}} \sum_{t=1}^{\overline{T}_{tv}} \delta_r(\tau)$. Now, given R independent runs, we collect all $\tilde{\delta}_r$ and define the "algorithm-related" mean optimality gap Cruz et al. (2011) and its related standard deviation. Table 2 shows the mean optimality gaps of each compared method on the test functions, with the standard deviations inside the parentheses. In both tables, we show in boldface the best means (we highlight multiple results in a row when they are within difference of 0.1).

We observe that for all considered problems, SMGO-Δ with the proposed time-varying model updates resulted in a very competitive performance compared with both of the BO-based approaches, when the means are considered. In the first three problems with $D = 3$, our method displayed much smaller standard deviations, pointing to a more consistent performance even with different starting optimization run conditions. Furthermore, SMGO-Δ showed the best mean optimality gaps for all problems with $D = 5$, while the standard deviations are competitive with those of the other methods.

Problem	**BO-R**	**BO-W**	**SMGO-Δ**
P1	3.94 (±1.45)	**3.59 (±2.29)**	4.07 (±0.90)
P2	**4.85 (±1.51)**	5.18 (±1.25)	5.06 (±1.03)
P3	**5.47 (±1.13)**	**5.45 (±1.06)**	**5.50 (±0.56)**
P4	6.42 (±0.67)	6.51 (±0.84)	**5.29 (±1.10)**
P5	6.41 (±0.69)	6.56 (±0.85)	**6.30 (±0.85)**
P6	6.72 (±0.80)	6.64 (±0.87)	**6.38 (±0.87)**

Table 2. Benchmark tests: mean and standard deviation of optimality gaps.

5.4 Comparative results: computational times

In the previous comparisons on optimization performance, we considered a generalized, unit-less time step in this test rather than wall-clock time, to ignore the effects of computational times of each algorithm, which may influence the sampling timings. Hence, for each time step in the same optimization run, all algorithms sample from the same (evolved) objective to maintain fairness. In this discussion, we now compare the computational times across the competing methods, noting that the tests were performed using MATLAB 2022a on a system with AMD Ryzen 9 3900X (3.80 GHz) and 32 GB RAM. We show in Figure 3 the spread of computational times required for each algorithm for every time step on Problem P1. Both BO-R and BO-W took higher computational times than SMGO, because for every new incoming sample, BO has to re-fit the required GP model on the updated data set. BO-W had a constantly high per-iteration computational time because of its constant size of samples set, i.e. for every oldest sample discarded, another new sample arrives. BO-R resulted in faster computing times than BO-W because of the regular clearing up of the data set after every batch. These problems can worsen significantly when we consider longer sliding window sizes for BO-W and larger batches for BO-R. On the other hand, SMGO had the best computing times, which, with the candidate points filtering method considered, has consistently kept most per-iteration times less than 0.2 s.

6. CONCLUSION

We proposed a new approach for time-varying black-box optimization, by using a Set Membership (SM) framework. To deal with the remembering-forgetting trade-off, we use the SM-based information from the new sample, and adjust the old model to agree with the new data. This adjustment renders old samples to be uninformative in the

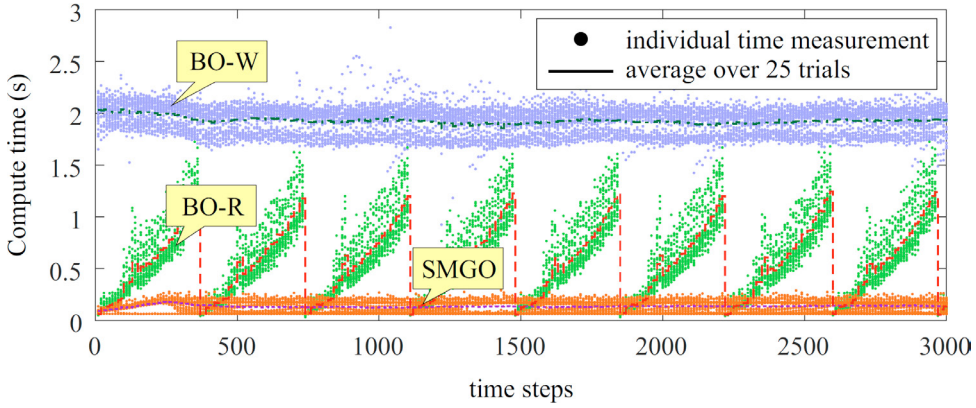

Figure 3. Comparison of computational times for the time-varying problem P1

sense of the tightness of their SM-based bounds, leading to their discarding from the data set. The updated model is then utilized for choosing the next sampling point, by automatically trading off between exploitation and exploration. The proposed method has low computational requirements and does not need any knowledge on the rate-of-change of the underlying function. The proposed technique is compared with BO-based methods on several benchmark tests in different problem dimensionalities. We demonstrated that the proposed method resulted in highly competitive performance, with better optimality gaps at five-dimensional benchmark problems, and with much shorter computational times.

REFERENCES

Antonio, C. (2019). Sequential model based optimization of partially defined functions under unknown constraints. *Journal of Global Optimization*, 79(2), 281–303.

Barrico, C. and Antunes, C.H. (2007). An evolutionary approach for assessing the degree of robustness of solutions to multi-objective models. In *Evolutionary Computation in Dynamic and Uncertain Environments*, 565–582. Springer.

Bemporad, A. (2020). Global optimization via inverse distance weighting and radial basis functions. *Computational Optimization and Applications*, 77(2), 571–595.

Blackwell, T. (2007). Particle swarm optimization in dynamic environments. *Evolutionary computation in dynamic and uncertain environments*, 29–49.

Blasi, L., Barbato, S., and Mattei, M. (2013). A particle swarm approach for flight path optimization in a constrained environment. *Aerospace Science and Technology*, 26(1), 128–137.

Bogunovic, I., Scarlett, J., and Cevher, V. (2016). Time-Varying Gaussian Process Bandit Optimization. In A. Gretton and C.C. Robert (eds.), *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, 314–323. PMLR, Cadiz, Spain.

Bosman, P.A. (2007). Learning and anticipation in online dynamic optimization. In *Evolutionary computation in dynamic and uncertain environments*, 129–152. Springer.

Cruz, C., González, J.R., and Pelta, D.A. (2011). Optimization in dynamic environments: a survey on problems, methods and measures. *Soft Computing*, 15(7), 1427–1448.

Fernandes, C.M., Lima, C., and Rosa, A.C. (2008). Umdas for dynamic optimization problems. In *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, GECCO '08, 399–406. Association for Computing Machinery, New York, NY, USA.

Finkel, D.E. and Kelley, C.T. (2006). Additive Scaling and the DIRECT Algorithm. *Journal of Global Optimization*, 36(4), 597–608.

Gelbart, M.A., Snoek, J., and Adams, R.P. (2014). Bayesian Optimization with Unknown Constraints. 1–14.

Hernández-Lobato, J.M., Gelbart, M.A., Adams, R.P., Hoffman, M.W., and Ghahramani, Z. (2016). A general framework for constrained bayesian optimization using information-based search. *J. Mach. Learn. Res.*, 17(1), 5549–5601.

Imamura, H., Charoenphakdee, N., Futami, F., Sato, I., Honda, J., and Sugiyama, M. (2020). Time-varying Gaussian Process Bandit Optimization with Non-constant Evaluation Time. *ArXiv*, abs/2003.04691.

Jones, D.R., Schonlau, M., and W. J. Welch (1998). Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global Optimization*, 13, 455–492.

Malherbe, C. and Vayatis, N. (2017). Global optimization of Lipschitz functions. *34th International Conference on Machine Learning, ICML 2017*, 5, 3592–3601.

Milanese, M. and Novara, C. (2004). Set Membership identification of nonlinear systems. *Automatica*, 40(6), 957–975.

Nyikosa, F.M., Osborne, M.A., and Roberts, S.J. (2018). Bayesian Optimization for Dynamic Problems.

Picheny, V., Trépos, R., and Casadebaig, P. (2017). Optimization of black-box models with uncertain climatic inputs—Application to sunflower ideotype design. *PLOS ONE*, 12(5), e0176815.

Sabug, L., Ruiz, F., and Fagiano, L. (2021). SMGO: A set membership approach to data-driven global optimization. *Automatica*, 133, 109890.

Sabug, L., Ruiz, F., and Fagiano, L. (2022). SMGO-Δ: Balancing Caution and Reward in Global Optimization with Black-Box Constraints. *Information Sciences*, 605, 15–42.

Silvas, E., Hofman, T., Murgovski, N., Etman, L.F., and Steinbuch, M. (2016). Review of Optimization Strategies for System-Level Design in Hybrid Electric Vehicles.

Yazdani, D., Omidvar, M.N., Cheng, R., Branke, J., Nguyen, T.T., and Yao, X. (2022). Benchmarking Continuous Dynamic Optimization: Survey and Generalized Test Suite. *IEEE Transactions on Cybernetics*, 52(5), 3380–3393.

Zhou, X. and Shroff, N. (2021). No-Regret Algorithms for Time-Varying Bayesian Optimization. In *2021 55th Annual Conference on Information Sciences and Systems (CISS)*, 1–6. IEEE.