Les Cahiers du GERAD ISSN: 0711–2440

STREGO: A scalable implementation of a trust-region framework for noisy Bayesian optimization

Y. Diouane, I. El Hallaoui, M. Kadri, A. Lamghari

G–2026–67

October 2026

La collection *Les Cahiers du GERAD* est constituée des travaux de recherche menés par nos membres. La plupart de ces documents de travail a été soumis à des revues avec comité de révision. Lorsqu'un document est accepté et publié, le pdf original est retiré si c'est nécessaire et un lien vers l'article publié est ajouté.

Citation suggérée : Y. Diouane, I. El Hallaoui, M. Kadri, A. Lamghari (Octobre 2026). STREGO: A scalable implementation of a trust-region framework for noisy Bayesian optimization, Rapport technique, Les Cahiers du GERAD G– 2026–67, GERAD, HEC Montréal, Canada.

Avant de citer ce rapport technique, veuillez visiter notre site Web (https://www.gerad.ca/fr/papers/G-2026-67) afin de mettre à jour vos données de référence, s'il a été publié dans une revue scientifique.

The series *Les Cahiers du GERAD* consists of working papers carried out by our members. Most of these pre-prints have been submitted to peer-reviewed journals. When accepted and published, if necessary, the original pdf is removed and a link to the published article is added.

Suggested citation: Y. Diouane, I. El Hallaoui, M. Kadri, A. Lamghari (October 2026). STREGO: A scalable implementation of a trust-region framework for noisy Bayesian optimization, Technical report, Les Cahiers du GERAD G–2026–67, GERAD, HEC Montréal, Canada.

Before citing this technical report, please visit our website (https://www.gerad.ca/en/papers/G-2026-67) to update your reference data, if it has been published in a scientific journal.

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

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

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

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

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

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

STREGO: A scalable implementation of a trust-region framework for noisy Bayesian optimization

Youssef Diouane [a, b]

Issmail El Hallaoui [a, b]

Meisseme Kadri [a, b]

Amina Lamghari [a, c]

[a] *GERAD, Montréal (Qc), Canada, H3T 1J4*

[b] *Département de mathématiques et génie industriel, Polytechnique Montréal, Montréal (Qc), Canada, H3T 0A3*

[c] *Département de management, Université du Québec à Trois-Rivières, Trois-Rivières (Qc), Canada, G8Z 4M3*

youssef.diouane@polymtl.ca
issmail.el-hallaoui@polymtl.ca
meisseme.kadri@etud.polymtl.ca
amina.lamghari@uqtr.ca

October 2026
Les Cahiers du GERAD
G–2026–67

Copyright © 2026 Diouane, El Hallaoui, Kadri, Lamghari

Les textes publiés dans la série des rapports de recherche *Les Cahiers du GERAD* n'engagent que la responsabilité de leurs auteurs. Les auteurs conservent leur droit d'auteur et leurs droits moraux sur leurs publications et les utilisateurs s'engagent à reconnaître et respecter les exigences légales associées à ces droits. Ainsi, les utilisateurs:

- Peuvent télécharger et imprimer une copie de toute publication du portail public aux fins d'étude ou de recherche privée;
- Ne peuvent pas distribuer le matériel ou l'utiliser pour une activité à but lucratif ou pour un gain commercial;
- Peuvent distribuer gratuitement l'URL identifiant la publication.

Si vous pensez que ce document enfreint le droit d'auteur, contactez-nous en fournissant des détails. Nous supprimerons immédiatement l'accès au travail et enquêterons sur votre demande.

The authors are exclusively responsible for the content of their research papers published in the series *Les Cahiers du GERAD*. Copyright and moral rights for the publications are retained by the authors and the users must commit themselves to recognize and abide the legal requirements associated with these rights. Thus, users:

- May download and print one copy of any publication from the public portal for the purpose of private study or research;
- May not further distribute the material or use it for any profit-making activity or commercial gain;
- May freely distribute the URL identifying the publication.

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Abstract : TREGO, a trust-region Bayesian optimization (BO) method for expensive blackbox problems, has convergence guarantees, but only for exact function values. In high dimension, its performance tends to deteriorate. This paper asks whether a trust-region BO method can be extended to handle noisy evaluations while being competitive in high dimension. We first generalize TREGO into a trust-region framework for noisy BO and extend its convergence analysis to noisy function estimates that are sufficiently accurate with sufficiently high probability. We then propose STREGO, a scalable implementation of this framework. Its global phase selects a batch of candidates by solving a bi-objective problem on the posterior mean of a Gaussian process and the integrated variance reduction. Unlike the predictive variance, this second objective does not favor boundary points merely because they are far from the data. Its local phase minimizes the posterior mean of a second Gaussian process, trained only on points close to the incumbent. STREGO is compared with TREGO, TURBO, BAXUS and NSMA-BO on synthetic and real-world problems of dimension 4 to 124, with and without noise. It gives the best average results in every synthetic setting tested and the best or tied-best solution on four of the six real-world problems. An implementation of STREGO is available at https://github.com/Mayousk/STREGO.

Keywords: Bayesian optimization; blackbox optimization; noisy derivative-free optimization; trust-region framework

Résumé : TREGO, une méthode d'optimisation bayésienne (BO) à région de confiance pour les problèmes boîte noire coûteux, bénéficie de garanties de convergence, mais uniquement pour des fonctions non bruitées. En grande dimension, ses performances ont tendance à se dégrader. Cet article examine si une méthode de BO à région de confiance peut être étendue pour traiter des évaluations bruitées tout en restant compétitive en grande dimension. Nous généralisons d'abord TREGO en un cadre à région de confiance pour la BO bruitée et étendons son analyse de convergence à des estimations bruitées de la fonction qui sont suffisamment précises avec une probabilité suffisamment élevée. Nous proposons ensuite STREGO, une instance de ce cadre général. Sa phase globale sélectionne un lot de candidats en résolvant un problème bi-objectif portant sur la moyenne a posteriori d'un processus gaussien et sur la réduction de variance intégrée. Contrairement à la variance prédictive, ce second objectif ne favorise pas les points de la frontière simplement parce qu'ils sont éloignés des données. Sa phase locale minimise la moyenne a posteriori d'un second processus gaussien, entraîné uniquement sur les points proches du meilleur point courant. STREGO est comparé à TREGO, TURBO, BAXUS et NSMA-BO sur des problèmes synthétiques et réels de dimension 4 à 124, avec et sans bruit. Il obtient les meilleurs résultats moyens dans tous les cadres synthétiques testés, ainsi que la meilleure solution, seul ou à égalité, sur quatre des six problèmes réels. Une implémentation de STREGO est disponible suivant ce lien https://github.com/Mayousk/STREGO.

Mots clés : Optimisation Bayésienne ; optimisation de boîtes noires ; optimisation sans dérivées bruitée ; méthodes à région de confiance

1 Introduction

In this paper, we consider the blackbox optimization problem

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

The objective function $f\colon \Omega \subseteq \mathbb{R}^d \to \mathbb{R}$ is locally Lipschitz continuous and bounded from below but cannot be evaluated directly. Optimization algorithms have access only to a stochastic zeroth-order oracle $f_\Theta(x)$, which returns a noisy observation of $f(x)$ and depends on a random variable Θ with an unknown distribution. The feasible set Ω is a closed convex set with a nonempty interior, defined by unrelaxable constraints [30], typically bound constraints. We focus on instances in which each call to the oracle is expensive and the dimension d is large, typically $d \geq 20$ [34].

Expensive calls limit the number of evaluations, and Bayesian optimization (BO) [20, 33, 45] is designed for this setting. It fits a probabilistic surrogate to the evaluations collected so far and selects the next point by maximizing a cheap acquisition function built on this surrogate. BO has become a standard tool for problems of the form (1) in machine learning [48], chemistry [22, 46] and aerospace design [41, 52]. Its best-known instance is efficient global optimization (EGO) [26], which pairs a Gaussian process (GP) [42] with the expected improvement (EI) [32]. Diouane et al. [16] embedded EGO in a trust-region scheme called TREGO. Each TREGO iteration starts with a global phase, which performs one EGO iteration over Ω. If this phase fails to achieve a sufficient decrease, a local phase maximizes EI inside a trust region around the incumbent. The sufficient-decrease condition, which also governs the trust-region radius, gives TREGO its convergence guarantees.

However, TREGO handles neither high dimension nor noise well. As d grows, the surrogate requires far more data to cover the domain, and the acquisition subproblem becomes harder to solve. Moreover, EI rewards a large predictive variance, which is a pointwise measure of uncertainty and is largest far from the observed data. In high dimension, the points farthest from the data typically lie near the boundary of Ω. Maximizing EI over the whole domain, as EGO and the global phase of TREGO do, therefore drives candidates toward the corners, where little is learned. This behavior is known as the boundary issue [36, 47]. With noise, the convergence guarantees of TREGO no longer apply, because they assume exact function values, which the oracle does not provide.

Convergence guarantees under noise have been established outside BO, for methods whose step acceptance relies, as in TREGO, on a sufficient-decrease condition. Chen et al. [11] proposed STORM, a trust-region method based on random models and on noisy function estimates that are sufficiently accurate with a fixed probability. They proved that STORM converges to a stationary point with probability one. Blanchet et al. [9] then used a supermartingale-based framework to bound the expected number of iterations STORM needs to reach an approximate stationary point. Paquette and Scheinberg [39] applied this probabilistic framework to stochastic line-search methods. Diouane et al. [15] extended it to evolution strategies, with a sufficient-decrease condition on noisy estimates and an extreme-barrier approach to constraints, and proved convergence to a first-order stationary point with probability one. More recently, Ding et al. [14] proposed a nonmonotone stochastic direct-search method and bounded its expected number of iterations with the stopping-time framework of [9]. All these methods, however, require the objective function to have a Lipschitz continuous gradient. For nonsmooth problems, Audet et al. [2] introduced StoMADS, which extends the mesh adaptive direct search (MADS) algorithm [1] and its convergence analysis to noisy estimates of the objective function. StoMADS requires only local Lipschitz continuity of the objective. Using the density of the generated directions and martingale theory, its authors proved convergence to a Clarke stationary point with probability one.

Within BO, most methods designed for high dimension are heuristic and come with no convergence guarantees. Several of them are built around trust regions [19, 35, 49], as is the local phase of TREGO. A trust region keeps candidates away from the boundary, and the surrogate only has to be accurate

inside it, which reduces the amount of data needed for a reliable fit. Some instead assume structure in the objective, such as additivity [5, 27, 54] or a low-dimensional active subspace [18, 38].

Other works modify the acquisition step. Batch methods select several points per iteration rather than one, to improve the exploration–exploitation balance and to use parallel computing resources [12, 21, 48]. Such batches are commonly produced by multistart strategies applied to the acquisition function. An alternative is to keep the posterior mean and variance of the surrogate as two separate objectives, rather than combine them into a single acquisition function, and to solve the resulting bi-objective problem [8, 53]. The Pareto front then offers a range of trade-offs between the two objectives and yields a batch naturally. Carciaghi et al. [10] built on this idea. They reconstruct the Pareto front with the Non-dominated Sorting Memetic Algorithm (NSMA) [29], which combines evolutionary operators with gradient-based refinement, and extract the batch by clustering, either in the input space or in the objective space. Their method is presented as a batch acquisition strategy and is explicitly studied for $d \geq 20$.

Each of these BO approaches has weaknesses. Trust-region solvers do not separate global exploration from local exploitation into two independent mechanisms. Methods based on structural assumptions fail, or waste evaluations, when the effective dimension is high. Multi-objective methods still use the predictive variance as their measure of uncertainty, so they remain exposed to the boundary issue. Clustering in the input space moves some candidates back toward the interior, but only as a correction applied after they are generated. In addition, most of these works assume noise-free evaluations. These observations raise the question that this paper addresses. Can a trust-region BO method keep its convergence guarantees with noisy evaluations and remain competitive in high dimension?

Our contributions are threefold. First, we generalize TREGO into a trust-region framework for noisy BO. Its definition and convergence analysis cover noisy function estimates that are sufficiently accurate with sufficiently high probability. They also cover iterations in which the trust region contains no feasible point, a case that TREGO does not address. As in StoMADS, the analysis requires f to be locally Lipschitz continuous rather than to have a Lipschitz continuous gradient. Second, we introduce STREGO, a scalable implementation of this framework that couples a bi-objective global infill criterion with a purely exploitative local phase. The two objectives of the global criterion are the posterior mean and the integrated variance reduction, which measures how much evaluating a candidate would reduce the predictive variance over the whole domain. A point near the boundary is therefore no longer favored merely because it is far from the observed data. The local phase minimizes, inside the trust region, the posterior mean of a separate GP trained only on the evaluated points near the incumbent, so that its lengthscales adapt to this region rather than to the whole domain. Third, we test STREGO on synthetic problems of dimension 4 to 100 and on real-world problems of dimension 16 to 124, with and without noise. A first experiment shows that, on three of the four synthetic functions tested, the integrated variance reduction generally leads to better solutions than the predictive variance. We then compare STREGO with TREGO and with TURBO [19], BAXUS [38] and NSMA-BO [10], three state-of-the-art solvers for high-dimensional BO. On the synthetic problems, STREGO obtains the best mean solution quality in every configuration tested. From dimension 50 on, it is also cheaper than all the other solvers, except NSMA-BO in the noise-free setting. On four of the six real-world problems, STREGO finds the best solution, alone on three and tied with TREGO on the fourth.

The remainder of the paper is organized as follows. Section 2 recalls EGO and TREGO, with a focus on the GP posterior. Section 3 presents the framework and its convergence analysis. Section 4 introduces STREGO, motivates the use of the integrated variance reduction and details the global and local phases. Section 5 reports the numerical experiments. Finally, Section 6 concludes the paper and outlines directions for future research. An implementation of STREGO is available at `https://github.com/Mayousk/STREGO`.

2 Preliminaries: EGO and TREGO

In this section, we first recall EGO, with a focus on the GP surrogate and the EI criterion and then describe the mechanism of TREGO.

2.1 EGO

EGO [26] is an instance of the BO framework in which a GP surrogate is trained on a growing design of experiments (DoE) and the next candidate is selected by maximizing the EI infill criterion. The DoE $\mathcal{D}_n = (\mathcal{X}_n, \mathcal{Y}_n)$ consists of n design points $\mathcal{X}_n = \{x_1, \dots, x_n\} \subset \Omega$ and their corresponding evaluations $\mathcal{Y}_n = \{y_1, \dots, y_n\}$ and the GP learns the mapping between the two. Formally, the objective function is modeled as a Gaussian process, $f \sim \mathcal{GP}(m, k)$, with constant prior mean m and covariance kernel k, whose hyperparameters are re-estimated from the collected data as the optimization progresses. Common choices are stationary kernels such as the squared-exponential kernel and the Matérn family. These kernels depend on the inputs only through the scaled distance $r(x, x') = \left(\sum_{i=1}^{d} (x_i - x'_i)^2/\ell_i^2\right)^{1/2}$. The lengthscale $\ell_i > 0$ controls how quickly the correlation between two points decays along the i-th coordinate. A small ℓ_i lets f vary rapidly in that direction, while a large ℓ_i makes it nearly flat.

Let $K \in \mathbb{R}^{n\times n}$ be the matrix with entries $K_{ij} = k(x_i, x_j)$ and $k_n(x) = [k(x, x_1), \dots, k(x, x_n)]^\top$. Conditioning on $\mathcal{D}_n$ yields a Gaussian posterior $f(x) \mid \mathcal{D}_n \sim \mathcal{N}\big(\mu_n(x), \sigma_n^2(x)\big)$ whose mean and variance are given in closed form by:

$$\mu_n(x) = m + k_n(x)^\top \left(K + \tau^2 I\right)^{-1} (\mathcal{Y}_n - m\mathbf{1}), \tag{2}$$

$$\sigma_n^2(x) = k(x, x) - k_n(x)^\top \left(K + \tau^2 I\right)^{-1} k_n(x), \tag{3}$$

where $\tau^2 \geq 0$ is the variance of the Gaussian observation noise ($\tau^2 = 0$ for noiseless evaluations), I is the $n \times n$ identity matrix and $\mathbf{1}$ is the n-dimensional vector of ones. The mean $\mu_n(x)$ is the model's estimate of $f(x)$, while the standard deviation $\sigma_n(x)$ quantifies the model's uncertainty about this estimate.

Given the current best value $y_n^{\min} = \min_{1\leq j\leq n} y_j$, EI measures the expected amount by which a candidate x improves upon it under the GP posterior, $\mathrm{EI}_n(x) = \mathbb{E}\big[\max\left(y_n^{\min} - f(x), 0\right) \big| \mathcal{D}_n\big]$. EI admits the closed form

$$\mathrm{EI}_n(x) = \begin{cases} \left(y_n^{\min} - \mu_n(x)\right) \Phi\big(z_n(x)\big) + \sigma_n(x)\, \phi\big(z_n(x)\big) & \text{if } \sigma_n(x) > 0, \\ \max\left(y_n^{\min} - \mu_n(x), 0\right) & \text{if } \sigma_n(x) = 0, \end{cases} \tag{4}$$

where $z_n(x) = \frac{y_n^{\min} - \mu_n(x)}{\sigma_n(x)}$ and Φ and ϕ denote the standard normal cumulative distribution and density functions. The first term rewards candidates with a low predicted mean (exploitation) and the second rewards candidates with high uncertainty (exploration).

2.2 TREGO

Building on the EGO framework and on the search-poll paradigm of direct-search methods [3], TREGO [16] embeds EGO within a trust-region scheme that comes with theoretical convergence guarantees. At each iteration k, TREGO maintains an incumbent x_k, its function value $f(x_k)$ and a trust-region radius $\delta_k > 0$. The iteration starts with a global phase and a local phase is performed only if the global phase fails to achieve a sufficient decrease.

The global phase performs one EGO iteration over the entire domain Ω: EI is maximized over Ω and the resulting candidate x_k^{global} is evaluated, yielding $f(x_k^{\text{global}})$. This candidate is accepted only if it achieves a sufficient decrease with respect to the incumbent, namely

$$f(x_k^{\text{global}}) \leq f(x_k) - \frac{\kappa}{2}\, \delta_k^2, \tag{5}$$

where $\kappa > 0$ is a user-defined constant. If (5) holds, the global phase is successful, the incumbent is updated to $x_{k+1} = x_k^{\text{global}}$, $\delta_{k+1} = \gamma\delta_k$ (with $\gamma > 1$) and the local phase is skipped. Otherwise, the iteration proceeds to the local phase.

The local phase follows the same mechanism, except that EI is maximized over a trust region around the incumbent given by $\overline{\Omega}_k$:

$$\overline{\Omega}_k = \{x \in \Omega \mid r_{\min}\,\delta_k \leq \|x - x_k\| \leq r_{\max}\,\delta_k\}\,, \tag{6}$$

where $0 < r_{\min} < r_{\max}$ are user-defined constants. The resulting local candidate $x_k^{\text{local}} \in \overline{\Omega}_k$ is evaluated and tested against the local sufficient decrease condition

$$f(x_k^{\text{local}}) \leq f(x_k) - \frac{\kappa}{2}\,\delta_k^2. \tag{7}$$

If this condition is satisfied, the local phase is a success, $x_{k+1} = x_k^{\text{local}}$ and the radius is expanded as $\delta_{k+1} = \gamma\,\delta_k$. Otherwise, the incumbent is kept, $x_{k+1} = x_k$ and the radius is shrunk as $\delta_{k+1} = \beta\,\delta_k$ with $0 < \beta < 1$.

The general framework of TREGO allows each phase to consist of several consecutive EGO steps: G steps over Ω in the global phase and L steps within $\overline{\Omega}_k$ in the local phase. The sufficient-decrease condition is checked only at the end of each phase, once all G (respectively L) steps have been performed. The resulting scheme is illustrated in Figure 1.

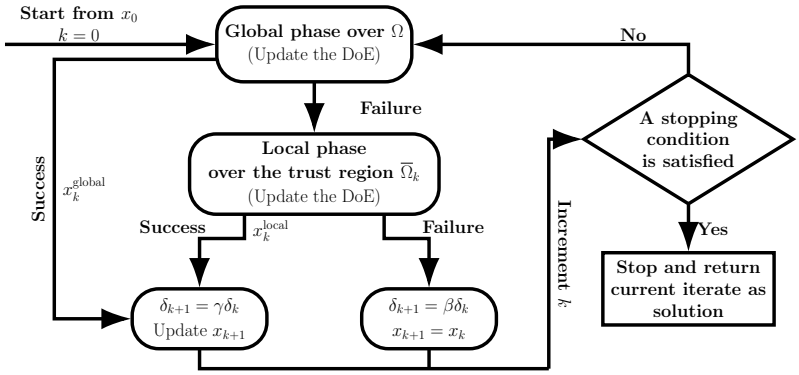


Figure 1: Overview of TREGO, adapted from [16] (here $0 < \beta < 1 < \gamma$).

3 A trust-region framework for noisy BO

In this section, we introduce a general stochastic trust-region framework for noisy BO. The framework is designed to be general enough to support different implementations. Its convergence analysis is given in Section 3.2.

3.1 The framework

We extend TREGO to the noisy setting, keeping the global and local steps unspecified. The way candidates are generated in each phase is left as a design choice and only the structure that alternates between the two phases and decides when a step is accepted is fixed.

Unlike TREGO, the proposed framework does not require exact function values at the global and local candidates. Instead, f_k^{global} and f_k^{local} are only assumed to be estimates of $f(x_k^{\text{global}})$ and $f(x_k^{\text{local}})$ and the framework leaves the estimation procedure open. Any estimator can therefore be used, ranging from a single noisy evaluation to a sample average over repeated evaluations at the same point, or the

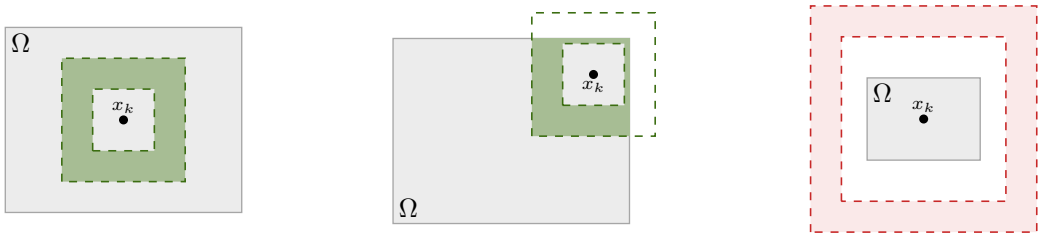
posterior mean of the surrogate. Furthermore, the incumbent value $f_k \approx f(x_k)$ is re-estimated at each iteration.

Once a global candidate is generated, the global sufficient-decrease condition given by Equation (5) becomes:

$$f_k^{\text{global}} \leq f_k - \frac{\kappa}{2}\delta_k^2, \tag{8}$$

If this condition is satisfied, the update proceeds as described in Section 2.2. Otherwise, the algorithm moves to the local phase.

The local phase is performed on a trust region $\overline{\Omega}_k$ defined as an annulus around the current incumbent x_k as stated in (6). However, in the constrained case where $\Omega \subsetneq \mathbb{R}^d$, this set may be empty. This might happen for instance when the step size δ_k is too large relative to the extent of the feasible domain around x_k. Figure 2 illustrates the possible configurations. In cases (a) and (b), $\overline{\Omega}_k$ is nonempty and the local phase is well defined. These are the only cases covered by TREGO [16]. Case (c), where $\overline{\Omega}_k = \emptyset$, is not addressed in the original framework. For this configuration, the local phase cannot be performed, so the iteration is not well defined and its convergence analysis does not apply. To close this gap, we introduce an additional (locally) empty iteration, in which the incumbent is kept and the step size is reduced. Since an empty trust region reveals nothing about f and is purely a geometric issue, the step size is reduced only mildly. The update rule is therefore given by $x_{k+1} = x_k$ and $\delta_{k+1} = \beta_0 \delta_k$ with $\beta_0 < 1$.



(a) $\overline{\Omega}_k \neq \emptyset$ (annulus inside). (b) $\overline{\Omega}_k \neq \emptyset$ (annulus partly outside). (c) $\overline{\Omega}_k = \emptyset$.

Figure 2: An $\mathbb{R}^2$ illustration of the annular trust region $\overline{\Omega}_k$ (using the infinity norm) for different step sizes and incumbent positions $x_k \in \Omega$.

When the trust region is nonempty ($\overline{\Omega}_k \neq \emptyset$), a local candidate x_k^{local} is generated within $\overline{\Omega}_k$. Its estimate f_k^{local} is then tested against the local sufficient-decrease condition:

$$f_k^{\text{local}} \leq f_k - \frac{\kappa}{2}\delta_k^2. \tag{9}$$

If this condition is satisfied, the iteration is successful, $x_{k+1} = x_k^{\text{local}}$ and $\delta_{k+1} = \gamma\delta_k$. In the original TREGO framework, if f_k^{local} does not satisfy this condition, the radius is shrunk by a parameter β. However, in the noisy setting, the estimates remain polluted by noise to some extent, so the outcome of the test is itself uncertain. Shrinking the trust region based on noisy estimates can therefore lead to a premature collapse. Instead, we distinguish two types of unsuccessful local iterations: certain and uncertain ones. If f_k^{local} sufficiently violates Equation (9) meaning,

$$f_k^{\text{local}} \geq f_k + \frac{\kappa}{2}\delta_k^2, \tag{10}$$

the iteration is declared a certain unsuccessful iteration, $x_{k+1} = x_k$ and the radius is shrunk by β_2 such that $\delta_{k+1} = \beta_2\delta_k$. Otherwise, the iteration is considered uncertainly unsuccessful, $x_{k+1} = x_k$ and the radius is shrunk by β_1 with $\beta_2 \leq \beta_1 \leq \beta_0 < 1$.

Commonly used in stochastic settings [2, 9], this procedure defines a tolerance interval within which the iteration is declared uncertainly unsuccessful and the trust region undergoes a mild shrinkage governed by β_1. Beyond this interval, the iteration is declared certainly unsuccessful and the radius is shrunk with the smaller parameter β_2. The complete procedure is detailed in Algorithm 1.

Algorithm 1: A trust-region framework for noisy BO.

Data: Initialize the step size parameter $\delta_0 > 0$, sample an initial DoE $\mathcal{D} = (\mathcal{X}, \mathcal{Y})$ of size N_0 with $\mathcal{X} = \{x_i\}_{i=1}^{N_0} \subset \Omega$, $\mathcal{Y} = \{f_i \approx f(x_i)\}_{i=1}^{N_0}$ and $x_0 \in \mathcal{X}$; Choose constants $0 < \beta_2 \leq \beta_1 \leq \beta_0 < 1 < \gamma$ such that $\gamma\beta_2 \geq 1$. Choose constants $\kappa > 0$, $r_{\max} > r_{\min} > 0$ and set $k = 0$.

while *the stopping criterion is not satisfied* **do**

Step 0 (estimate at the incumbent):

Compute an estimate $f_k \approx f(x_k)$.

Step 1 (global BO phase over Ω):

Generate a global candidate $x_k^{\text{global}} \in \Omega$ and an estimate $f_k^{\text{global}} \approx f(x_k^{\text{global}})$.

Step 2 (global sufficient-decrease test):

if $f_k^{\text{global}} \leq f_k - \frac{\kappa}{2}\delta_k^2$ **then**

The iteration is (globally) successful. Set $x_{k+1} = x_k^{\text{global}}$ and $\delta_{k+1} = \gamma\delta_k$.

else

Define the local trust region $\overline{\Omega}_k = \{x \in \Omega \mid r_{\min}\,\delta_k \leq \|x - x_k\| \leq r_{\max}\,\delta_k\}$.

if $\overline{\Omega}_k \neq \emptyset$ **then**

Step 3 (local BO phase over the trust region $\overline{\Omega}_k$):

Generate a local candidate $x_k^{\text{local}} \in \overline{\Omega}_k$ and an estimate $f_k^{\text{local}} \approx f(x_k^{\text{local}})$.

Step 4 (local sufficient-decrease test):

if $f_k^{\text{local}} \leq f_k - \frac{\kappa}{2}\delta_k^2$ **then**

The iteration is (locally) successful. Set $x_{k+1} = x_k^{\text{local}}$ and $\delta_{k+1} = \gamma\delta_k$.

else

if $f_k^{\text{local}} \geq f_k + \frac{\kappa}{2}\delta_k^2$ **then**

The iteration is a certain unsuccessful iteration. Set $x_{k+1} = x_k$ and $\delta_{k+1} = \beta_2\delta_k$.

else

The iteration is an uncertain unsuccessful iteration. Set $x_{k+1} = x_k$ and $\delta_{k+1} = \beta_1\delta_k$.

end

end

else

The iteration is (locally) empty. Set $x_{k+1} = x_k$ and $\delta_{k+1} = \beta_0\delta_k$.

end

end

Increment k;

end

3.2 Convergence analysis

Throughout this section, the objective function f is assumed to be accessed only through stochastic estimators. As a consequence, all the quantities generated by Algorithm 1 are considered as random variables defined on a probability space $(\mathcal{E}, \mathcal{G}, \mathbb{P})$: the iterate X_k, the step-size parameter Δ_k, the direction Q_k, the step $S_k = \Delta_k Q_k$ and the estimates F_k and F_k^s of f at X_k and at the trial point $X_k + S_k$ defined below. Given a sample $\omega \in \mathcal{E}$, we denote their realizations by lowercase letters: $x_k = X_k(\omega)$, $\delta_k = \Delta_k(\omega)$, $q_k = Q_k(\omega)$, $s_k = S_k(\omega)$, $f_k = F_k(\omega)$ and $f_k^s = F_k^s(\omega)$. We denote by $\mathcal{F}_{k-1}$ the σ-algebra generated by all the random quantities produced by Algorithm 1 up to iteration $k-1$, in particular by $F_0, F_0^s, \ldots, F_{k-1}, F_{k-1}^s$, with $\mathcal{F}_{-1} = \sigma(x_0)$. Thus, $\{\mathcal{F}_k\}_{k\geq -1}$ is a filtration and X_k and Δ_k are $\mathcal{F}_{k-1}$-measurable.

Note that Algorithm 1 can be interpreted as generating, at each iteration k, a trial point

$$x_k^s = \begin{cases} x_k^{\text{local}} & \text{if the global phase is declared unsuccessful and } \overline{\Omega}_k \neq \emptyset, \\ x_k^{\text{global}} & \text{otherwise.} \end{cases} \tag{11}$$

The point x_k^s is always involved in the last sufficient-decrease test performed at iteration k. By construction, $x_k^s \in \Omega$ for all k. Since $x_0 \in \Omega$ and $x_{k+1} \in \{x_k, x_k^s\}$, all the iterates also belong to Ω, so that $f(X_k)$ and $f(X_k + S_k)$ are well defined. We denote by $\mathcal{U}$ the (random) set of indices k for which the global phase is declared unsuccessful and $\overline{\Omega}_k \neq \emptyset$ (the iterations where the local phase over $\overline{\Omega}_k$ is performed) and by $\mathcal{U}_\emptyset$ the (random) set of indices for which the global phase is declared unsuccessful and $\overline{\Omega}_k = \emptyset$. The stochastic convergence analysis is inspired by what is done in [2, 11, 39].

In order to obtain convergence guarantees for Algorithm 1, we require the estimates of f to be sufficiently accurate with a suitable probability. This is formalized in Definition 3.1.

Definition 3.1. Given constants $\varepsilon_f > 0$ and $p \in (0, 1]$, the sequence of the random quantities F_k and F_k^s is called p-probabilistically ε_f-accurate, for corresponding sequences $\{X_k\}$, $\{S_k\}$, $\{\Delta_k\}$, if the event

$$T_k^f \triangleq \left\{ |F_k - f(X_k)| \leq \varepsilon_f \Delta_k^2 \text{ and } |F_k^s - f(X_k + S_k)| \leq \varepsilon_f \Delta_k^2 \right\}$$

satisfies the condition

$$p_k^* \triangleq \mathbb{P}\left(T_k^f \mid \mathcal{F}_{k-1} \right) \geq p.$$

We derive a convergence analysis of Algorithm 1 under the following assumptions.

Assumption 3.1. f is bounded below by f_{low} on Ω.

Assumption 3.2. The sequence of random objective function estimates $\{F_k, F_k^s\}_k$ generated by Algorithm 1 satisfies the following two conditions.

(1) The sequence $\{F_k, F_k^s\}_k$ is p-probabilistically ε_f-accurate for some $p \in (\frac{1}{2}, 1]$ and $\varepsilon_f \in (0, \frac{\kappa}{8}]$, where κ is the constant used in Algorithm 1.

(2) There exists $\varepsilon_v > 0$ such that the sequence of estimates $\{F_k, F_k^s\}_k$ satisfies the following ε_v-variance condition for all $k \geq 0$,

$$\mathbb{E}\left(|F_k - f(X_k)|^2 \mid \mathcal{F}_{k-1} \right) \leq \varepsilon_v^2 \Delta_k^4 \quad \text{and}$$
$$\mathbb{E}\left(|F_k^s - f(X_k + S_k)|^2 \mid \mathcal{F}_{k-1} \right) \leq \varepsilon_v^2 \Delta_k^4.$$

For the sake of our analysis, we introduce the following (random) Lyapunov function

$$\Phi_k = \nu f(X_k) + (1 - \nu)\Delta_k^2, \tag{12}$$

where $\nu \in (0, 1)$ satisfies

$$\frac{\nu}{1 - \nu} \geq \frac{8}{\kappa} \max\{1 - \beta_0^2, \gamma^2 - 1\}. \tag{13}$$

Since $f(x_0)$ is finite, $f \geq f_{\text{low}}$ on Ω and, on successful iterations, $f(X_{k+1}) \leq f(X_k) + |F_k^s - f(X_k + S_k)| + |F_k - f(X_k)|$, where the error terms are integrable by Assumption 3.2(2) and $\Delta_k \leq \gamma^k \delta_0$, an induction on k shows that Φ_k is integrable, so all conditional expectations below are well defined. Consider a realization of Algorithm 1 and let ϕ_k be the corresponding realization of Φ_k. We denote by S the event "the iteration k is successful", by $\overline{S}$ its complement and by $\mathbf{1}_S$ (respectively $\mathbf{1}_{\overline{S}}$) the corresponding indicator function.

Lemma 3.1. Let Assumptions 3.1 and 3.2 hold. Then, for every iteration k, one has

$$\max\Big[\mathbb{E}\left(\mathbf{1}_{\overline{T_k^f}} |F_k - f(X_k)| \mid \mathcal{F}_{k-1} \right), \\ \mathbb{E}\left(\mathbf{1}_{\overline{T_k^f}} |F_k^s - f(X_k + S_k)| \mid \mathcal{F}_{k-1} \right) \Big] \leq (1 - p_k^*)^{1/2} \varepsilon_v \Delta_k^2 \leq (1 - p)^{1/2} \varepsilon_v \Delta_k^2,$$

where $\overline{T_k^f}$ denotes the event where the objective function estimates (F_k, F_k^s) are inaccurate and $\mathbf{1}_{\overline{T_k^f}}$ is the indicator function of the event set $\overline{T_k^f}$.

Proof. The result follows from the conditional Cauchy–Schwarz inequality, $\mathbb{E}(\mathbf{1}_{\overline{T_k^f}} | \mathcal{F}_{k-1}) = 1 - p_k^*$ and Assumption 3.2 (2) as done in [39]. □

The next lemma quantifies the decrease of the Lyapunov function on the event where the estimates are accurate.

Lemma 3.2. Let Assumptions 3.1 and 3.2 hold and let ν satisfy (13). Then, for every iteration k, one has

$$\mathbb{E}\left(\mathbf{1}_{T_k^f} (\Phi_{k+1} - \Phi_k) \mid \mathcal{F}_{k-1} \right) \leq -(1 - \nu)(1 - \beta_0^2) p_k^* \Delta_k^2 \leq -(1 - \nu)(1 - \beta_0^2) p \, \Delta_k^2,$$

where $\mathbf{1}_{T_k^f}$ is the indicator function of the event set T_k^f.

Proof. Consider a realization of a given iteration k of Algorithm 1 for which the objective function estimates $\{f_k, f_k^s\}$ are ε_f-accurate. If the iteration is unsuccessful, then $x_{k+1} = x_k$ and $\delta_{k+1} \leq \max\{\beta_2, \beta_1, \beta_0\}\, \delta_k = \beta_0\, \delta_k$, since $\beta_2 \leq \beta_1 \leq \beta_0 < 1$, which leads to

$$\phi_{k+1} - \phi_k \leq (1-\nu)(\beta_0^2 - 1)\delta_k^2 := b_1 < 0. \tag{14}$$

Otherwise, if the iteration is successful, then one has $x_{k+1} = x_k^s = x_k + \delta_k q_k$, $f_k^s - f_k \leq -\frac{\kappa}{2}\delta_k^2$ and $\delta_{k+1} = \gamma\delta_k$. One thus has

$$\begin{aligned}
\phi_{k+1} - \phi_k &= \nu\,(f(x_{k+1}) - f(x_k)) + (1-\nu)(\gamma^2 - 1)\delta_k^2 \\
&\leq \left(\nu\left(2\varepsilon_f - \frac{\kappa}{2}\right) + (1-\nu)(\gamma^2-1)\right)\delta_k^2 \\
&\leq \left(-\nu\frac{\kappa}{4} + (1-\nu)(\gamma^2-1)\right)\delta_k^2,
\end{aligned}$$

where the last inequality is obtained using the fact that $\varepsilon_f \leq \frac{\kappa}{8}$. Using (13), we have $\frac{\nu}{1-\nu} \geq \frac{8(\gamma^2-1)}{\kappa}$, which implies

$$\phi_{k+1} - \phi_k \leq -\nu\frac{\kappa}{8}\delta_k^2 := b_2 < 0.$$

We note also that (13) gives $\frac{\nu}{1-\nu} \geq \frac{8}{\kappa}(1-\beta_0^2)$ and hence $b_2 \leq b_1$. Hence, one deduces that for any iteration k, on this event, one has

$$\mathbf{1}_{T_k^f}\,(\Phi_{k+1} - \Phi_k) \leq -\mathbf{1}_{T_k^f}(1-\nu)(1-\beta_0^2)\Delta_k^2.$$

Taking the conditional expectation with respect to $\mathcal{F}_{k-1}$ one gets

$$\mathbb{E}\left(\mathbf{1}_{T_k^f}\,(\Phi_{k+1} - \Phi_k)\,\middle|\mathcal{F}_{k-1}\right) \leq -(1-\nu)(1-\beta_0^2)p_k^*\Delta_k^2 \leq -(1-\nu)(1-\beta_0^2)p\,\Delta_k^2. \qquad \square$$

We now turn to the event where the estimates are inaccurate, on which the Lyapunov function may increase.

Lemma 3.3. Let Assumptions 3.1 and 3.2 hold and let ν satisfy (13). Then, for every iteration k, one has

$$\mathbb{E}\left(\mathbf{1}_{\overline{T_k^f}}(\Phi_{k+1} - \Phi_k)\middle|\mathcal{F}_{k-1}\right) \leq 2\nu(1-p_k^*)^{1/2}\varepsilon_v\Delta_k^2 \leq 2\nu(1-p)^{1/2}\varepsilon_v\Delta_k^2,$$

where $\mathbf{1}_{\overline{T_k^f}}$ is the indicator function of the event set $\overline{T_k^f}$, i.e., the indicator over the set of inaccurate estimates (F_k, F_k^s).

Proof. Conditioned on $\mathcal{F}_{k-1}$, assume that the objective function estimates $\{F_k, F_k^s\}$ are inaccurate. If the iteration k is successful, one has $X_{k+1} = X_k + S_k$, $F_k^s - F_k \leq -\frac{\kappa}{2}\Delta_k^2$ and $\Delta_{k+1} = \gamma\Delta_k$. Hence,

$$\begin{aligned}
f(X_{k+1}) - f(X_k) &= (f(X_k + S_k) - F_k^s) + (F_k^s - F_k) + (F_k - f(X_k)) \\
&\leq |f(X_k + S_k) - F_k^s| + |F_k - f(X_k)| - \frac{\kappa}{2}\Delta_k^2,
\end{aligned}$$

which leads to

$$\begin{aligned}
\mathbf{1}_{\overline{T_k^f}}\mathbf{1}_S(\Phi_{k+1} - \Phi_k) \leq \mathbf{1}_{\overline{T_k^f}}\mathbf{1}_S\Big(&\nu\,|f(X_k + S_k) - F_k^s| + \nu\,|F_k - f(X_k)| \\
&+ \left(-\nu\frac{\kappa}{2} + (1-\nu)(\gamma^2-1)\right)\Delta_k^2\Big).
\end{aligned}$$

Using (13), we have $(1-\nu)(\gamma^2-1) \leq \nu\frac{\kappa}{8} \leq \nu\frac{\kappa}{2}$. Hence, one gets

$$\mathbf{1}_{\overline{T_k^f}}\mathbf{1}_S(\Phi_{k+1} - \Phi_k) \leq \mathbf{1}_{\overline{T_k^f}}\mathbf{1}_S\,\nu\,(|f(X_k + S_k) - F_k^s| + |F_k - f(X_k)|)\,. \tag{15}$$

If the iteration is unsuccessful (certain, uncertain or empty), then $X_{k+1} = X_k$ and $\Delta_{k+1} \leq \beta_0 \Delta_k$, which leads to

$$\mathbf{1}_{\overline{T_k^f}} \mathbf{1}_{\overline{S}} (\Phi_{k+1} - \Phi_k) \leq -\mathbf{1}_{\overline{T_k^f}} \mathbf{1}_{\overline{S}} (1-\nu)(1-\beta_0^2)\Delta_k^2 \leq 0. \tag{16}$$

Since the right-hand side of (15) is nonnegative, combining (15) and (16) yields, in both cases,

$$\mathbf{1}_{\overline{T_k^f}} (\Phi_{k+1} - \Phi_k) \leq \mathbf{1}_{\overline{T_k^f}} \nu \left(|f(X_k + S_k) - F_k^s| + |F_k - f(X_k)| \right).$$

Taking the conditional expectation with respect to $\mathcal{F}_{k-1}$ and applying Lemma 3.1, one obtains

$$\mathbb{E}\left(\mathbf{1}_{\overline{T_k^f}} (\Phi_{k+1} - \Phi_k) \middle| \mathcal{F}_{k-1} \right) \leq 2\nu(1-p_k^*)^{1/2} \varepsilon_v \Delta_k^2 \leq 2\nu(1-p)^{1/2} \varepsilon_v \Delta_k^2. \qquad \square$$

Putting Lemmas 3.2 and 3.3 together yields the expected decrease of the Lyapunov function, provided that the estimates are accurate with a sufficiently high probability.

Theorem 3.1. Let Assumptions 3.1 and 3.2 hold, where the probability p satisfies

$$4\nu\varepsilon_v(1-p)^{1/2} \leq (1-\nu)(1-\beta_0^2)\, p, \tag{17}$$

and let $\nu \in (0,1)$ satisfy (13). Then, there exists an $\alpha > 0$ such that, for all k,

$$\mathbb{E}\left(\Phi_{k+1} - \Phi_k \middle| \mathcal{F}_{k-1} \right) \leq -\alpha \Delta_k^2. \tag{18}$$

Proof. Putting the results of the two Lemmas 3.2 and 3.3 together, we obtain

$$\begin{aligned} \mathbb{E}\left(\Phi_{k+1} - \Phi_k \middle| \mathcal{F}_{k-1} \right) &= \mathbb{E}\left((\mathbf{1}_{T_k^f} + \mathbf{1}_{\overline{T_k^f}})(\Phi_{k+1} - \Phi_k) \middle| \mathcal{F}_{k-1} \right) \\ &\leq \left(-(1-\nu)p(1-\beta_0^2) + 2\nu(1-p)^{1/2}\varepsilon_v \right) \Delta_k^2. \end{aligned}$$

Hence, using (17), it reduces to

$$\mathbb{E}\left(\Phi_{k+1} - \Phi_k \middle| \mathcal{F}_{k-1} \right) \leq -\frac{1}{2}(1-\nu)p(1-\beta_0^2)\Delta_k^2 = -\alpha\Delta_k^2,$$

where $\alpha = \frac{1}{2}(1-\nu)p(1-\beta_0^2) > 0$. $\square$

Hence, Theorem 3.1 ensures that the Lyapunov function decreases over iterations in expectation, as long as the accuracy probability of the estimates of f is high enough. Using such a result, one can guarantee that the whole sequence of step sizes $\{\Delta_k\}$ converges to zero almost surely.

Corollary 3.1. Suppose that the working assumptions of Theorem 3.1 are satisfied. Then, the sequence $\{\Delta_k\}_k$ goes to 0 almost surely. Moreover, if the sequence of iterates $\{X_k\}$ is bounded, then there exist a subsequence $\mathcal{K}$ and a random vector $X_* \in \Omega$ such that $\{\Delta_k\}_{k\in\mathcal{K}}$ goes to zero and $\{X_k\}_{k\in\mathcal{K}}$ converges almost surely to $X_* \in \Omega$.

Proof. By taking expectation on the result from Theorem 3.1, one gets

$$\begin{aligned} \mathbb{E}\left(\Phi_{n+1} - \Phi_0 \right) &= \sum_{k=0}^{n} \mathbb{E}\left(\Phi_{k+1} - \Phi_k \right) = \sum_{k=0}^{n} \mathbb{E}\left(\mathbb{E}\left(\Phi_{k+1} - \Phi_k \middle| \mathcal{F}_{k-1} \right) \right) \\ &\leq -\alpha \mathbb{E}\left(\sum_{k=0}^{n} \Delta_k^2 \right). \end{aligned}$$

Since $X_{n+1} \in \Omega$, one has $\Phi_{n+1} \geq \nu f_{\text{low}}$ and by taking $n \to \infty$,

$$\mathbb{E}\left(\sum_{k=0}^{+\infty} \Delta_k^2\right) < \infty.$$

Thus, we conclude that the probability of the random variable Δ_k converging to zero is one. Moreover, assuming the boundedness of the sequence $\{X_k\}$, one deduces the existence of a random vector X_* and a subsequence $\mathcal{K} \subset \mathbb{N}$ such that $\{\Delta_k\}_{k\in\mathcal{K}}$ goes to zero almost surely and $\{X_k\}_{k\in\mathcal{K}} \subset \Omega$ converges almost surely to $X_* \in \Omega$ (as Ω is a closed set). □

Corollary 3.1 shows that the event $A_\Delta = \{\lim_{k\to\infty} \Delta_k = 0\}$ is almost sure. On this event, the trust region eventually contains at least one point, so empty iterations can only occur finitely often. We now state this result formally.

Proposition 3.1. Let the assumptions of Corollary 3.1 hold. Then, for every $\omega \in A_\Delta$, one has $\overline{\Omega}_k \neq \emptyset$ for all k sufficiently large. In particular, the set $\mathcal{U}_\emptyset$ of empty iterations is finite almost surely.

Proof. Let $\omega \in A_\Delta$. Since Ω has a nonempty interior, $\operatorname{diam}(\Omega) > 0$ and, by the triangle inequality, $\sup_{y\in\Omega} \|y - x\| \geq \frac{1}{2}\operatorname{diam}(\Omega)$ for every $x \in \Omega$. As $\delta_k \to 0$, there exists k_1 such that $r_{\min}\delta_k < \frac{1}{2}\operatorname{diam}(\Omega)$ for all $k \geq k_1$. For such k, there exists $\bar{y} \in \Omega$ with $\|\bar{y} - x_k\| > r_{\min}\delta_k$. By convexity of Ω, the point

$$y = x_k + \frac{r_{\min}\delta_k}{\|\bar{y} - x_k\|}(\bar{y} - x_k)$$

lies on the segment $[x_k, \bar{y}] \subseteq \Omega$ and satisfies $\|y - x_k\| = r_{\min}\delta_k \leq r_{\max}\delta_k$. Hence $y \in \overline{\Omega}_k$ and $\overline{\Omega}_k \neq \emptyset$ for all $k \geq k_1$. Consequently, $\mathcal{U}_\emptyset(\omega) \subseteq \{0, \ldots, k_1 - 1\}$. □

Before stating our main result, we recall an auxiliary result from the literature [17] that will be useful for the analysis.

Lemma 3.4. Let $\{G_k\}$ be a submartingale with respect to a filtration $\{\mathcal{F}_k\}$, i.e., $\mathbb{E}(G_k|\mathcal{F}_{k-1}) \geq G_{k-1}$ for all k. Assume further that there exists $M > 0$ such that $|G_k - G_{k-1}| \leq M < \infty$, for all k. Consider the random events $C = \{\lim_{k\to\infty} G_k \text{ exists and is finite}\}$ and $D = \{\limsup_{k\to\infty} G_k = \infty\}$. Then $\mathbb{P}(C \cup D) = 1$.

For $k \in \mathcal{U}$, we define the random variable Ψ_k with realizations

$$\psi_k = \frac{f(x_k) - f(x_k + \delta_k q_k)}{\delta_k}, \qquad k \in \mathcal{U}.$$

Lemma 3.5. Let the working assumptions of Corollary 3.1 hold. Then, almost surely, the set $\mathcal{U}$ of iterations where the local phase is performed is infinite and

$$\liminf_{k\in\mathcal{U}} \Psi_k \leq 0.$$

Proof. Let $Z_k = \sum_{i=0}^{k} \left(2\mathbf{1}_{T_i^f} - 1\right)$, where T_i^f is given in Definition 3.1 and recall that, by Assumption 3.2, $p_i^* = \mathbb{P}(T_i^f|\mathcal{F}_{i-1}) \geq p > \frac{1}{2}$. Since Z_{k-1} is $\mathcal{F}_{k-1}$-measurable, $\{Z_k\}$ is a submartingale:

$$\mathbb{E}(Z_k|\mathcal{F}_{k-1}) = Z_{k-1} + 2\mathbb{P}(T_k^f|\mathcal{F}_{k-1}) - 1 \geq Z_{k-1}.$$

Note that $|Z_k - Z_{k-1}| = 1$, hence the event $\{\lim_{k\to\infty} Z_k \text{ exists and is finite}\}$ is empty. Thus, by Lemma 3.4, one deduces that the event $\{\limsup_k Z_k = \infty\}$ is almost sure. Recall also, from Corollary 3.1, that the event A_Δ is almost sure.

Let $\epsilon > 0$ be a rational number and consider the event

$$E_\epsilon = \{\omega \in \mathcal{E} : \exists k' \in \mathbb{N} \text{ such that } \Psi_k(\omega) \geq \epsilon \text{ for all } k \in \mathcal{U}(\omega),\ k \geq k'\}.$$

We show that $E_\epsilon \cap A_\Delta \cap \{\limsup_k Z_k = \infty\} = \emptyset$. We proceed by contradiction and consider ω in this intersection, with the corresponding realizations $x_k = X_k(\omega)$, $\delta_k = \Delta_k(\omega)$, $q_k = Q_k(\omega)$, $\psi_k = \Psi_k(\omega)$ and $z_k = Z_k(\omega)$. Since $\delta_k \to 0$ and by Proposition 3.1, there exists $k_0 \geq k'$ such that, for all $k \geq k_0$, $k \notin \mathcal{U}_\emptyset$ and $\delta_k \leq b_\epsilon := \frac{2\epsilon}{\kappa + 4\varepsilon_f}$.

Let B_k be the random variable whose realization is $b_k = \log_\gamma\left(\frac{\delta_k}{b_\epsilon}\right)$; thus, $b_k \leq 0$ for all $k \geq k_0$. For any $k \geq k_0$, no empty iteration occurs and three cases remain.

- If $k \notin \mathcal{U}$, then, since $k \notin \mathcal{U}_\emptyset$, the global phase is successful, $\delta_{k+1} = \gamma\delta_k$ and $b_{k+1} = b_k + 1$.
- If $k \in \mathcal{U}$ and $\mathbf{1}_{T_k^f} = 1$, then $x_k^s = x_k^{\text{local}} = x_k + \delta_k q_k$ and $f(x_k + \delta_k q_k) - f(x_k) = -\delta_k\psi_k \leq -\epsilon\delta_k$. Hence, since $\delta_k \leq b_\epsilon$,
$$f_k^s - f_k \leq 2\varepsilon_f\delta_k^2 - \epsilon\delta_k \leq -\frac{\kappa}{2}\delta_k^2.$$
Hence, the iteration is successful, $\delta_{k+1} = \gamma\delta_k$ and $b_{k+1} = b_k + 1$.
- If $k \in \mathcal{U}$ and $\mathbf{1}_{T_k^f} = 0$, then, since $\delta_{k+1} \geq \beta_2\delta_k$ and $\gamma\beta_2 \geq 1$, one has $b_{k+1} \geq b_k + \log_\gamma(\beta_2) \geq b_k - 1$.

In all cases, $b_{k+1} - b_k \geq 2\mathbf{1}_{T_k^f}(\omega) - 1 = z_k - z_{k-1}$ for all $k \geq k_0$. Summing these inequalities, one gets $b_k - b_{k_0} \geq z_{k-1} - z_{k_0-1}$ for all $k > k_0$. Since $\limsup_k z_k = \infty$, one obtains $\limsup_k b_k = \infty$, which contradicts the fact that $b_k \leq 0$ for all $k \geq k_0$. Hence $E_\epsilon \cap A_\Delta \cap \{\limsup_k Z_k = \infty\} = \emptyset$ and since A_Δ and $\{\limsup_k Z_k = \infty\}$ are almost sure, $\mathbb{P}(E_\epsilon) = 0$.

Let $A_\Psi = \mathcal{E} \setminus \bigcup_{\epsilon \in \mathbb{Q}_{>0}} E_\epsilon$, which is almost sure as the complement of a countable union of null events. Consider $\omega \in A_\Psi$. If $\mathcal{U}(\omega)$ were finite, then, taking $k' > \max\mathcal{U}(\omega)$, one would have $\omega \in E_\epsilon$ for every ϵ, which is impossible. Hence $\mathcal{U}(\omega)$ is infinite. Moreover, for every rational $\epsilon > 0$ and every k', there exists $k \in \mathcal{U}(\omega)$, $k \geq k'$, such that $\psi_k < \epsilon$, i.e., $\liminf_{k \in \mathcal{U}(\omega)} \psi_k \leq 0$. □

Lemma 3.5 shows that, almost surely, the local phase is performed infinitely often and fails to produce sufficient decrease along some subsequence of these iterations. The analysis can therefore focus on the local phase iterations, and in particular on the limit points they generate. This leads to the notions of refining subsequences and refining directions associated with the local phase.

Definition 3.2. Consider a realization of Algorithm 1. A subsequence of iterates $\{x_k\}_{k \in \mathcal{K}}$, with $\mathcal{K} \subseteq \mathcal{U}$ infinite, is called a *refining subsequence* if $\lim_{k \in \mathcal{K}} \delta_k = 0$ and $\{x_k\}_{k \in \mathcal{K}}$ converges to some $x_* \in \Omega$, called a *refined point*. A vector $q \in \mathbb{R}^d$ is a *refining direction* for x_*, associated with $\mathcal{K}$, if there exists an infinite subset $\mathcal{K}' \subseteq \mathcal{K}$ such that $\lim_{k \in \mathcal{K}'} q_k = q$.

For $k \in \mathcal{K} \subseteq \mathcal{U}$, the local phase is performed at iteration k, so that $x_k^s = x_k^{\text{local}} \in \overline{\Omega}_k$. By definition of $\overline{\Omega}_k$ and of q_k, one has $\|q_k\| = \|x_k^{\text{local}} - x_k\|/\delta_k$ and hence $r_{\min} \leq \|q_k\| \leq r_{\max}$. Therefore, refining directions always exist and since this annulus is closed, any refining direction q satisfies $r_{\min} \leq \|q\| \leq r_{\max}$. Finally, since the iterates are bounded by assumption, refining subsequences exist whenever $\mathcal{K}$ is infinite and $\{\delta_k\}_{k \in \mathcal{K}}$ converges to zero.

Since we are in the constrained case, $y + tq$ is not guaranteed to belong to Ω for y close to x_* and small $t > 0$, so the classical Clarke generalized derivative cannot be used. We now introduce the tools needed to state our final convergence result for the constrained setting.

Definition 3.3. A vector $q \in \mathbb{R}^d$ is a *hypertangent vector* to Ω at $x \in \Omega$ if there exists $\varepsilon > 0$ such that

$$y + tu \in \Omega \qquad \forall\, y \in \Omega \cap B(x;\varepsilon),\ u \in B(q;\varepsilon),\ 0 < t < \varepsilon.$$

The set of such vectors, denoted by $T_\Omega^H(x)$, is an open cone called the *hypertangent cone*.

When f is Lipschitz continuous relative to Ω near $x \in \Omega$, the Clarke–Jahn generalized derivative of f at x in the direction $q \in T_\Omega^H(x)$ is defined by

$$f^\circ(x;q) := \limsup_{\substack{y \to x,\ y \in \Omega \\ t \downarrow 0,\ y+tq \in \Omega}} \frac{f(y+tq)-f(y)}{t}. \tag{19}$$

We now state the main global convergence result for Algorithm 1.

Theorem 3.2. Let the working assumptions of Corollary 3.1 hold, and assume also that the sequence of iterates $\{X_k\}$ generated by Algorithm 1 is almost surely bounded. Then, there exists an almost sure event A such that, for all $\omega \in A$, there exists a refining subsequence $\{X_k(\omega)\}_{k \in \mathcal{K}(\omega)}$ associated with the local phase, with refined point $X_*(\omega) \in \Omega$, such that $\lim_{k \in \mathcal{K}(\omega)} \Psi_k(\omega) \leq 0$. Moreover, any refining direction $Q(\omega)$ for $X_*(\omega)$ satisfies $r_{\min} \leq \|Q(\omega)\| \leq r_{\max}$ and, if $Q(\omega) \in T_\Omega^H\big(X_*(\omega)\big)$,

$$f^\circ\left(X_*(\omega); Q(\omega)\right) \geq 0.$$

Proof. From Corollary 3.1, Lemma 3.5 and the fact that the sequence $\{X_k\}$ is almost surely bounded, the event $A = A_\Delta \cap A_\Psi \cap \{\{X_k\} \text{ is bounded}\}$ is almost sure as the intersection of three almost sure events. Now, consider $\omega \in A$ and let $x_k = X_k(\omega)$, $\delta_k = \Delta_k(\omega)$, $q_k = Q_k(\omega)$ and $\psi_k = \Psi_k(\omega)$. By Lemma 3.5, there exists an infinite subset $\mathcal{K}_0 \subseteq \mathcal{U}$ such that $\lim_{k \in \mathcal{K}_0} \psi_k = \liminf_{k \in \mathcal{U}} \psi_k \in [-\infty, 0]$. Since $\{x_k\}_{k \in \mathcal{K}_0}$ is bounded, there exists an infinite subset $\mathcal{K} = \mathcal{K}(\omega) \subseteq \mathcal{K}_0$ such that $\{x_k\}_{k \in \mathcal{K}}$ converges to some x_*, which belongs to Ω since Ω is closed. As $\omega \in A_\Delta$, one also has $\lim_{k \in \mathcal{K}} \delta_k = 0$, so that $\{x_k\}_{k \in \mathcal{K}}$ is a refining subsequence with refined point x_*, and $\lim_{k \in \mathcal{K}} \psi_k = \lim_{k \in \mathcal{K}_0} \psi_k \leq 0$.

Since f is locally Lipschitz continuous, there exist $r > 0$ and $\lambda > 0$ such that f is Lipschitz continuous with constant λ on $\Omega \cap B(x_*; r)$. As $x_k \to x_*$, $\delta_k \to 0$ and $\|q_k\| \leq r_{\max}$ for $k \in \mathcal{U}$, the points x_k and $x_k + \delta_k q_k = x_k^{\text{local}} \in \overline{\Omega}_k \subseteq \Omega$ belong to $\Omega \cap B(x_*; r)$ for $k \in \mathcal{K}$ sufficiently large. Hence $|\psi_k| \leq \lambda \|q_k\| \leq \lambda r_{\max}$ for such k, and $\lim_{k \in \mathcal{K}} \psi_k$ is finite.

Since $\{q_k\}_{k \in \mathcal{K}}$ lies in the compact annulus $\{q : r_{\min} \leq \|q\| \leq r_{\max}\}$, refining directions for x_* associated with $\mathcal{K}$ exist and all satisfy $r_{\min} \leq \|q\| \leq r_{\max}$. Let q be any such refining direction, and let $\mathcal{K}' \subseteq \mathcal{K}$ be an infinite subset such that $\lim_{k \in \mathcal{K}'} q_k = q$.

Assume now that $q \in T_\Omega^H(x_*)$ and let $\varepsilon > 0$ be given by Definition 3.3 for q at x_*. Since $x_k \to x_*$ and $\delta_k \to 0$, one has, for $k \in \mathcal{K}'$ sufficiently large, $x_k \in \Omega \cap B(x_*; \varepsilon)$ and $0 < \delta_k < \varepsilon$; applying Definition 3.3 with $y = x_k$, $u = q$ and $t = \delta_k$ gives $x_k + \delta_k q \in \Omega$. Moreover, since $\|q\| \leq r_{\max}$, one also has $x_k + \delta_k q \in B(x_*; r)$ for $k \in \mathcal{K}'$ sufficiently large. Therefore, for $k \in \mathcal{K}'$ sufficiently large, the three points x_k, $x_k + \delta_k q_k$ and $x_k + \delta_k q$ belong to $\Omega \cap B(x_*; r)$ and, by Lipschitz continuity of f on $\Omega \cap B(x_*; r)$,

$$\frac{f(x_k+\delta_k q)-f(x_k)}{\delta_k} = \frac{f(x_k+\delta_k q)-f(x_k+\delta_k q_k)}{\delta_k} + \frac{f(x_k+\delta_k q_k)-f(x_k)}{\delta_k} \geq -\lambda\|q-q_k\| - \psi_k.$$

Since $y = x_k \in \Omega$, $y \to x_*$, $t = \delta_k \downarrow 0$ and $y + tq = x_k + \delta_k q \in \Omega$ for $k \in \mathcal{K}'$ sufficiently large, these points are admissible in (19) and one gets

$$f^\circ(x_*; q) \geq \limsup_{k \in \mathcal{K}'} \frac{f(x_k+\delta_k q)-f(x_k)}{\delta_k} \geq \limsup_{k \in \mathcal{K}'} \left(-\psi_k - \lambda\|q-q_k\|\right) = -\lim_{k \in \mathcal{K}} \psi_k \ \geq \ 0,$$

where the equality uses $\lim_{k \in \mathcal{K}'} \|q - q_k\| = 0$ and the fact that $\{\psi_k\}_{k \in \mathcal{K}'}$ is a subsequence of the convergent sequence $\{\psi_k\}_{k \in \mathcal{K}}$. □

If, in addition, the set of normalized refining directions is dense in $T_\Omega^H(X_*(\omega)) \cap \{u : \|u\| = 1\}$, then $f^\circ(X_*(\omega); q) \geq 0$ for all $q \in T_\Omega^H(X_*(\omega))$, meaning $X_*(\omega)$ is a Clarke stationary point of f over Ω [1].

4 STREGO: A scalable implementation of Algorithm 1

In this section, we introduce STREGO, a scalable instance of the generalized framework defined in Section 3 tailored to noisy, high-dimensional blackbox problems. To this end, we design a batch-based global phase coupled with an exploitative local phase. We first describe the overall procedure, then detail the global and local phases. STREGO begins by sampling an initial set of points $\{x_i\}_{i=1}^{N_0}$ and evaluating their objective values. We denote the initial sample set by $\mathcal{X} = \{x_i\}_{i=1}^{N_0}$ and the corresponding evaluations by $\mathcal{Y} = \{f_i \approx f(x_i)\}_{i=1}^{N_0}$. Together, the pairs $\{(x_i, f_i)\}_{i=1}^{N_0}$ define the initial DoE $\mathcal{D}$. We also define a starting point x_0. The algorithm then alternates between a global and a local phase. The global phase consists of training a GP on the entire DoE and solving a bi-objective problem based on the posterior information. Each global iteration produces a batch of G points, denoted $\mathcal{X}^{\mathrm{G}}$, which are evaluated. The global solution x_k^{global} is the best point among the G candidates (where k is the number of completed global-local steps). If the global phase is unsuccessful (according to (8)), the algorithm proceeds to the local search within a trust region around the incumbent (6). In this phase, the local candidate x_k^{local} minimizes the posterior mean of a local surrogate model. We now detail the global phase and the local phase.

4.1 The global phase of STREGO

The global phase starts by building and training a GP on the entire DoE. Two quantities are obtained from the model: the posterior mean μ and the posterior variance σ^2. The mean serves as an estimate of the objective function, while the variance represents the model's uncertainty about this estimate. Recall that EGO suggests the next candidate by the scalar acquisition function EI. However, as shown in [13], EI tends to suggest points that have already been evaluated or that lie close to previously evaluated ones. Another difficulty appears in high dimension, where EI tends to propose points on the boundary of the search space. This is known as the boundary issue [36, 47]. Indeed, σ^2 is a purely local measure. It quantifies the model's uncertainty at x itself, independent of how that uncertainty relates to the rest of Ω. This locality is the root of the boundary issue. Near the boundary of Ω, the kernel's correlation neighborhood is truncated by the domain edge. Since no observations can lie on the other side, boundary points receive less information than interior points. This inflates their variance relative to points near the center and the effect worsens with dimension. Because variance-based acquisition criteria reward high σ^2 directly, they are repeatedly drawn toward these boundary regions, regardless of whether these regions are actually the most informative. Moreover, a scalar acquisition function captures only one exploration–exploitation trade-off.

To address this, some works directly sample from the Pareto front of a bi-objective [13, 25], or more generally multi-objective [31], problem built on the posterior of the surrogate. More specifically, Carciaghi et al. [10] formulate this as a mean–variance bi-objective problem, whose infill criterion is given by:

$$\min_{x \in \Omega} \big(\mu(x), -\sigma^2(x)\big). \tag{20}$$

Although separating the mean and variance gives access to diverse trade-offs, ranking candidates by σ^2 does not mitigate the boundary issue. In parallel, several works have developed stepwise uncertainty reduction (SUR) strategies for GP surrogates [6, 40]. At each step, the next candidate is chosen to minimize the expected remaining uncertainty about a quantity of interest once it is observed. Within this framework, taking the posterior variance as the uncertainty measure recovers the integrated mean squared error (IMSE) criterion of Sacks et al. [43], originally a one-shot design criterion for globally accurate surrogates. More recently, it was applied sequentially in active learning, where it is known as the integrated mean squared prediction error (IMSPE) [7]. The same quantity appears in the literature as integrated variance reduction (IVR) and as the negative integrated posterior variance up to an additive constant. Rather than ranking a candidate solely by its current predictive variance, IVR evaluates it according to the expected reduction in posterior variance over a reference domain. Therefore, it favors observations whose influence extends across a large portion of Ω.

Formally, given the current GP posterior with predictive variance σ^2, the integrated variance reduction of a candidate x is defined as:

$$\mathrm{IVR}(x) = \int_{\Omega} \left[\sigma^2_{\mathcal{D}}(x') - \sigma^2_{\mathcal{D}\cup\{x\}}(x')\right] \pi(x')\, dx', \tag{21}$$

where $\sigma^2_{\mathcal{D}}$ and $\sigma^2_{\mathcal{D}\cup\{x\}}$ denote the posterior variance of the surrogate before and after adding x to the current set of observations $\mathcal{D}$, and π is a reference measure over Ω. Thus, IVR measures how much querying x would reduce the uncertainty of the model over the entire domain Ω, rather than the uncertainty at x alone. This distinction is central in the expensive setting considered here. Under a fixed evaluation budget, each evaluation should improve the model where it matters for the search, not only at the candidate itself. By favoring candidates whose information propagates across the domain, IVR extracts more information from each evaluation and it does not reward boundary points merely because they are isolated. Note that (21) depends on the data only through the design locations $\mathcal{X}$, not through the observed responses $\mathcal{Y}$. In practice we approximate π by a fixed reference set and evaluate the resulting criterion in closed form. Further details are given in Section 5.2.

In this context, our proposed bi-objective problem for the global phase is given by

$$\min_{x\in\Omega} \big(\mu(x), -\mathrm{IVR}(x)\big). \tag{22}$$

Solving this bi-objective problem yields an approximation of the Pareto set $\mathcal{X}^{\mathrm{P}} \subseteq \Omega$. This set typically contains many more points than the batch size G, so only a subset of them can be evaluated. Choosing this subset poorly may result in candidates that carry essentially the same information, thereby wasting costly evaluations. Both the predictive mean and the IVR criterion are derived from similarity kernels defined in terms of input-space distances. Consequently, low-mean candidates lie near previously well-performing observations, while high-IVR candidates lie in undersampled regions, and these regions are typically well separated in the input space. Following this reasoning, and as done in [10], we generate the batch of G global candidates by clustering the approximate Pareto set $\mathcal{X}^{\mathrm{P}}$ in the input space. Given a target batch size G, we divide the Pareto set points into G clusters. A representative candidate is then selected from each cluster to ensure diversity. The resulting batch is set as $\mathcal{X}^{\mathrm{G}} = \{x_i^{\mathrm{G}}\}_{i=1}^{\mathrm{G}}$. The DoE is augmented with the G evaluated solutions and the global solution x_k^{global} is computed as the best point among them in terms of function value.

Selecting G candidates at once allows them to be evaluated in parallel and gives a better coverage of the search space, which is difficult to achieve in high dimension. The main steps are reported in Algorithm 2. Given the corresponding input, it generates a batch of G points, evaluates them and returns both the augmented DoE and the best point of the batch. In STREGO, that point is the global candidate x_k^{global} used in Step 1 of Algorithm 1.

Algorithm 2: Step 1 of Algorithm 1.

Data: Oracle function f, feasible domain Ω, batch size $\mathrm{G} \geq 1$, DoE $\mathcal{D} = (\mathcal{X}, \mathcal{Y})$.
Result: x^{global}, f^{global} and $\mathcal{D}$.

1: Fit a GP surrogate on $\mathcal{D}$ with posterior mean μ and IVR.
2: Find a Pareto set $\mathcal{X}^{\mathrm{P}}$ for the bi-objective problem: $\min_{x\in\Omega}\big(\mu(x), -\mathrm{IVR}(x)\big)$.
3: Select a batch $\mathcal{X}^{\mathrm{G}} = \{x_i\}_{i=1}^{\mathrm{G}} \subset \mathcal{X}^{\mathrm{P}}$ (e.g., via a clustering method).
4: Estimate the selected batch $\mathcal{Y}^{\mathrm{G}} = \{f_i \approx f(x_i)\}_{i=1}^{\mathrm{G}}$.
5: Let x^{global} be the best estimated point in the batch and $f^{\mathrm{global}} \approx f(x^{\mathrm{global}})$.
6: Update the DoE $\mathcal{D}$, i.e., $\mathcal{X} \leftarrow \mathcal{X} \cup \mathcal{X}^{\mathrm{G}}$ and $\mathcal{Y} \leftarrow \mathcal{Y} \cup \mathcal{Y}^{\mathrm{G}}$.

4.2 The local phase of STREGO

The global phase explores the domain to generate promising starting points for local refinement. Because it must balance exploration and exploitation, the bi-objective formulation is well suited to this role.

The local phase, by contrast, is purely exploitative. Once a promising region has been identified, the task is no longer to explore but to refine the estimate of the optimum within that region. Consequently, we discard the variance term and rely solely on the predictive mean inside the trust region:

$$\min_{x\in\overline{\Omega}} \mu(x), \tag{23}$$

where $\overline{\Omega}$ is the local trust region around a given incumbent point x_{inc}

$$\overline{\Omega} = \{x \in \Omega \mid r_{\min}\,\delta \le \|x - x_{\text{inc}}\| \le r_{\max}\,\delta\}\,, \tag{24}$$

where δ is the trust-region radius and $0 < r_{\min} < r_{\max}$ are user-defined constants.

In TREGO's original implementation, the GP surrogate is trained on the entire DoE, which conflicts with the local nature of the trust-region logic, especially in high-dimensional settings. This choice is problematic for two reasons. First, fitting a single stationary kernel to the entire landscape forces one lengthscale to capture the behavior near the incumbent and the global structure of the objective. As the trust region shrinks, this mismatch can cause the surrogate to degenerate [19, 49]. Second, GP training scales cubically with the number of points. Since the DoE grows by G points every global iteration, retraining on the full DoE becomes increasingly expensive. Therefore, we adopt an independent local model for the local phase and restrict its training set to the points lying within the trust region. This keeps the fitting cost bounded and allows the lengthscale to adapt to the local geometry of $\overline{\Omega}$ rather than to the whole domain. Problem (23) becomes

$$\min_{x\in\overline{\Omega}} \overline{\mu}(x), \tag{25}$$

where $\overline{\mu}$ is a local surrogate trained only using the local DoE. However, the trust region may contain too few evaluated points to train an accurate local model. We therefore train it on the relaxed local DoE

$$\overline{\mathcal{X}} = \{x \in \mathcal{X} \mid \|x - x_{\text{inc}}\| \le r_{\max}\,\rho\delta\}\,, \tag{26}$$

where $\rho \ge 1$ is a relaxation factor chosen so that the size of the local DoE $\overline{\mathcal{X}}$ is above a certain threshold $n_{\min}$. Relaxing the trust region enlarges the pool of already evaluated points available to the local model, providing more training data and improving its accuracy in the neighborhood of the incumbent. The local candidate x_k^{local} is then obtained by solving (25), where the local GP is trained on the whole local DoE $\overline{\mathcal{X}}$. Given the current DoE, incumbent x_k and δ_k, Algorithm 3 generates the local candidate x_k^{local} and updates the DoE with the new evaluation.

Algorithm 3: Step 3 of Algorithm 1.

Data: Oracle function f, incumbent x_{inc}, trust-region radius $\delta > 0$, constants $r_{\max} > r_{\min} > 0$, a size threshold $n_{\min} > 1$ and the DoE $\mathcal{D} = (\mathcal{X}, \mathcal{Y})$.
Result: x^{local}, f^{local} and $\mathcal{D}$.

1: Define the relaxed local DoE

$$\overline{\mathcal{X}} = \{x \in \mathcal{X} \mid \|x - x_k\| \le \rho\, r_{\max}\delta_k\},$$

where $\rho \ge 1$ is the smallest value such that $|\overline{\mathcal{X}}| \ge n_{\min}$. Set $\overline{\mathcal{Y}} = \{f_i : x_i \in \overline{\mathcal{X}}\}$.
2: Fit a local GP on $\overline{\mathcal{D}} = (\overline{\mathcal{X}}, \overline{\mathcal{Y}})$ and let $\overline{\mu}$ denote its posterior mean.
3: Minimize the posterior mean $\overline{\mu}$ over the local trust region, as defined in (24), i.e.,

$$x^{\text{local}} \in \operatorname*{argmin}_{x\in\overline{\Omega}} \overline{\mu}(x) \quad \text{and} \quad f^{\text{local}} \approx f(x^{\text{local}}).$$

4: Update the DoE $\mathcal{D}$, i.e., $\mathcal{X} \leftarrow \mathcal{X} \cup \{x^{\text{local}}\}$ and $\mathcal{Y} \leftarrow \mathcal{Y} \cup \{f^{\text{local}}\}$.

5 Numerical experiments

The experiments in this section serve two purposes. The first is to assess the choice of IVR, rather than the predictive variance, as the uncertainty measure of the global phase. The second is to compare STREGO with TURBO, BAXUS, NSMA-BO and TREGO.

5.1 Experimental setting

We first use five synthetic benchmark functions, described in Appendix A together with the noise models and the shift procedure. Two are separable (Alpine-01, Schwefel), two are nonseparable (Ackley, Rotated Rastrigin) and one, Hartmann-6, has a low-dimensional structure that a solver can exploit. Each function is tested in dimensions $d \in \{4, 20, 50, 100\}$, without noise and with two types of Gaussian noise. The additive noise has a standard deviation of approximately 10% of the function's output range in the dimension considered, and the multiplicative noise a relative standard deviation of 0.1. Hartmann-6 requires $d \geq 6$ and is therefore not tested at $d = 4$. To avoid favoring certain solvers, we shift the functions whose optimum lies at the origin or near the boundary of the feasible domain. We then complete the comparison on real-world engineering problems from the blackbox benchmark Bencher [37].

5.2 Implementation details

STREGO was implemented in Python 3 and the experiments were run on a Linux machine with an Intel 12th Gen Core i7-12700 (x86_64) CPU, using 6 CPU cores.

The GP of each phase is a `SingleTaskGP` model from BoTorch [4]. It uses an unscaled radial basis function (RBF) kernel with automatic relevance determination (one lengthscale ℓ_i per coordinate) and the dimensionality-scaled lengthscale prior of Hvarfner et al. [24].

In the global phase, we approximate IVR (21) by the average variance reduction over 64 reference points spread evenly across the domain by a Sobol sequence. These points are generated once and reused throughout, so that all candidates are compared on the same basis. The global bi-objective problem is solved with NSMA [29], with a population of 100 individuals and at most 20 generations. Its operators are simulated binary crossover (SBX), with probability 0.9 and distribution index $\eta_c = 20$, and polynomial mutation, with distribution index $\eta_m = 20$. A crowding-based mechanism, with a crowding quantile of 0.9, promotes diversity along the Pareto front, and a gradient-based refinement step is performed every $n_{\text{opt}} = 5$ iterations. The other settings of NSMA are its defaults.

We select the batch from the approximate Pareto set returned by NSMA, using k-means clustering in the input space with $k = \text{G}$ and $n_{\text{init}} = 10$ random initializations. Because the centroid of a cluster is the mean of its Pareto-set points and need not be one of them, we retain, for each cluster, the Pareto-set point nearest to its centroid. We set $\text{G} = 3$ because three clusters tend to yield one candidate with a low posterior mean, one with a high IVR and one compromise between the two. Appendix C shows that $\text{G} = 3$ is preferable in high dimension, although $\text{G} = 1$ performs better at $d = 4$.

In the local phase, the local GP is trained on the relaxed local DoE (26), with $\rho \geq 1$ chosen as the smallest value such that $|\overline{\mathcal{X}}| \geq n_{\min} = 10$. Its posterior mean is minimized over the annulus (24), with $r_{\min}$ set to the machine precision, since Algorithm 1 requires $r_{\min} > 0$. The minimization uses the `optimize_acqf` routine of BoTorch, with the posterior mean as acquisition function, $(2d + 4)$ restarts and $(2d + 4)^2$ raw samples drawn by Sobol sampling. For large d, the number of raw samples can be reduced to limit memory use.

The evaluation budget is limited to 200, which reflects the expensive setting considered in this paper. Although the candidates in a batch could be evaluated in parallel, we evaluate them sequentially. Each benchmark is run over 20 independent trials, with all solvers started from the same initial DoE of size $N_0 = 10$. STREGO is run with the hyperparameters summarized in Table B1 of Appendix B and each baseline with the default hyperparameters recommended in its original paper.

5.3 Metrics

We use two metrics. The first, f_{best}, is the best objective value found so far. Convergence curves show its mean over the 20 trials as a function of the number of evaluations, with shaded regions indicating

the 95% confidence interval around the mean. The second, f_{final}, is the best objective value a solver reaches within the budget on a given benchmark. To compare problems of different scales, we normalize f_{final} for each function and dimension,

$$\text{Normalized } f_{\text{final}} = \frac{f_{\text{final}} - f_{\min}}{f_{\max} - f_{\min}},$$

where $f_{\min}$ and $f_{\max}$ are the best and worst values observed across all solvers and all trials on that function at that dimension. A normalized value of 0 thus corresponds to the best value of the group, and a value of 1 to the worst. In the noisy settings, all reported values are the noisy values observed by the optimizer rather than the underlying true objective, so that the reported performance reflects what an algorithm could actually measure in practice. We also report the CPU time needed to first reach f_{final}, excluding the cost of the initial design.

The normalized values of f_{final} are displayed as violin plots, one per solver. A violin gathers one value per benchmark function and trial, at a fixed dimension and noise setting. Its width is a kernel density estimate of these values, so that the violin is wide where many runs concentrate and narrow elsewhere. The central bar marks the median, and the upper and lower bands mark the worst and best runs. A violin concentrated near 0 therefore indicates a solver whose final values are close to the best of the group across functions and trials, whereas a bimodal shape indicates a solver whose behavior differs from one function to another.

5.4 Variance versus IVR

We compare two variants of STREGO that differ only in the uncertainty objective of the global bi-objective problem: the predictive variance, as in problem (20), and IVR, as in problem (22). We refer to them as $(\mu, -\sigma^2)$ and $(\mu, -\text{IVR})$, respectively. Both use the local phase described in Section 4.2. Figure 3 shows the evolution of f_{best} for both variants on the four functions tested at every dimension.

On Ackley, Alpine-01 and Rotated Rastrigin, IVR generally yields better final values than the predictive variance, for d ranging from 4 to 100. For instance, on the noise-free Ackley function at $d = 50$, the final value is ≈ 17.3 with IVR, as opposed to ≈ 19.6 with the variance. On Rotated Rastrigin under additive noise at $d = 100$, the values are ≈ 1000 and ≈ 1120, respectively. The gap is smaller on Ackley under multiplicative noise at $d = 20$ and $d = 100$, where the two variants end close to each other. Schwefel is the exception: the two criteria perform comparably, and the variance-based variant is slightly better in some cases, in particular under multiplicative noise at $d = 20$ (≈ 5100 as opposed to ≈ 5350).

5.5 Comparison with state-of-the-art solvers

We now compare STREGO with the four solvers named at the beginning of this section. TURBO [19] and BAXUS [38] are well-established baselines for high-dimensional BO and handle the dimension in different ways. TURBO confines the search to one or several trust regions, runs an independent BO process in each and allocates evaluations among them through an implicit multi-armed bandit. BAXUS restricts the dimensionality of the search rather than its region: it optimizes over a nested sequence of random linear embeddings, expands the embedding whenever progress stalls and uses a TURBO-style trust region within each subspace. NSMA-BO [10] solves problem (20) of Section 4.1, and the results reported in [10] show that it is also competitive in high dimension. Finally, TREGO [16], described in Section 2.2, is included because STREGO builds on it, so the comparison evaluates the effectiveness of the features that STREGO adds. We use the implementation of TREGO in egobox [28], within the Surrogate Modeling Toolbox (SMT) [44]. This implementation uses the same GP in the global and local phases and EI as the acquisition function in both.

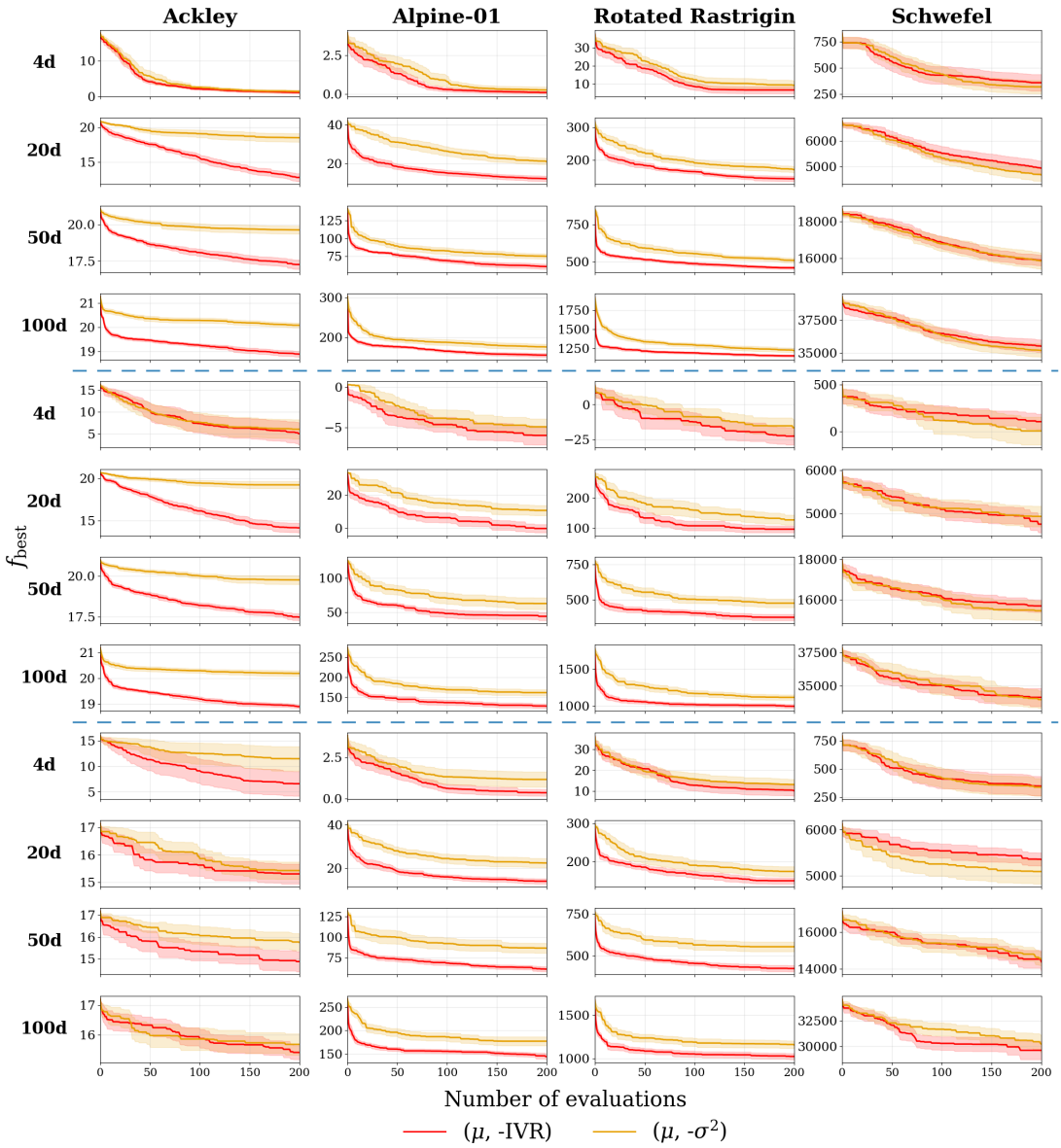
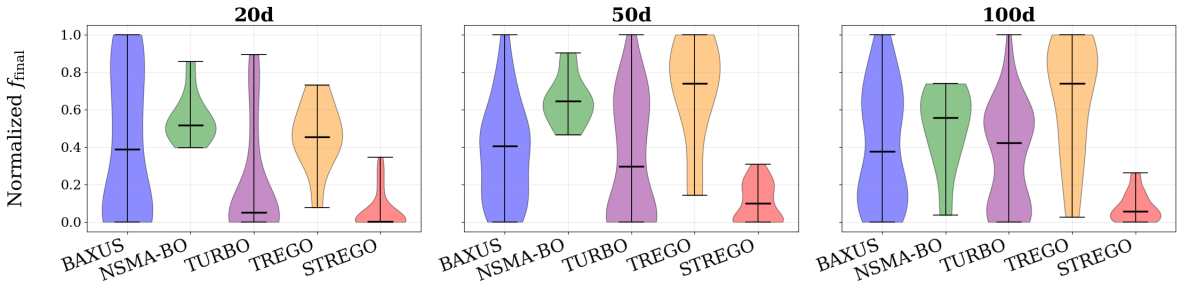


Figure 3: Comparison of STREGO with the global bi-objective criterion $(\mu, -\text{IVR})$ against its variance counterpart $(\mu, -\sigma^2)$, both using the exploitative local phase, on Ackley, Alpine-01, Rotated Rastrigin and Schwefel in 4d, 20d, 50d and 100d under different noise settings.

5.5.1 Results on synthetic benchmarks

We separate the effect of the dimension from that of the noise. Figure 4 varies the dimension in the noise-free setting, and Figures 5 and 6 vary the noise setting at $d = 4$ and $d = 50$, respectively. Each figure shows the violins of the normalized f_{final}, with tables of the mean normalized f_{final} and of the mean CPU time needed to first reach it.

STREGO obtains the best mean normalized f_{final} in every configuration of Figures 4 to 6, between 0.061 and 0.195, and the most compact violins. Its lead over the next best solver is largest under additive noise at $d = 4$, where the other four solvers obtain means between 0.474 and 0.511, as opposed to 0.149 for STREGO. The ranking of the other solvers depends on the dimension and on the noise. In the noise-free setting, TURBO ranks second at every dimension, with means between 0.136 and



Normalized final objective value

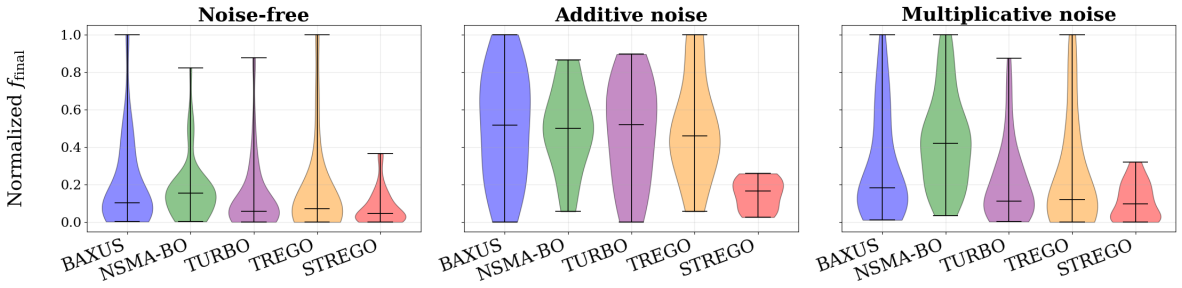
Solver	20d	50d	100d
BAXUS	0.408 ± 0.049	0.376 ± 0.038	0.413 ± 0.042
NSMA-BO	0.550 ± 0.046	0.652 ± 0.041	0.499 ± 0.054
TURBO	0.200 ± 0.041	0.339 ± 0.046	0.354 ± 0.036
TREGO	0.462 ± 0.050	0.694 ± 0.040	0.653 ± 0.049
STREGO	$\mathbf{0.061} \pm 0.018$	$\mathbf{0.107} \pm 0.015$	$\mathbf{0.080} \pm 0.010$

CPU time (s)

Solver	20d	50d	100d
BAXUS	253.23 ± 13.44	609.87 ± 27.65	689.12 ± 55.23
NSMA-BO	475.95 ± 3.23	$\mathbf{372.56} \pm 11.50$	$\mathbf{503.46} \pm 14.18$
TURBO	$\mathbf{63.78} \pm 2.92$	515.66 ± 24.66	559.66 ± 18.58
TREGO	176.83 ± 8.50	913.53 ± 9.25	6271.72 ± 88.22
STREGO	334.48 ± 25.43	449.19 ± 32.99	518.80 ± 29.79

Figure 4: Normalized final objective value (lower is better) and CPU time for $d = 20$, 50 and 100 in the noise-free setting. Tables report the mean $\pm$ standard error over 20 trials and 5 functions.

0.354, and BAXUS ranks third for $d \geq 20$ (0.376 to 0.413). TREGO degrades as the dimension grows. It ranks third at $d = 4$ (0.165), fourth at $d = 20$ (0.462) and last at $d = 50$ and $d = 100$ (0.694 and 0.653), where its runs cover most of $[0, 1]$. This degradation is consistent with the two limitations of TREGO discussed in Sections 4.1 and 4.2, namely the attraction of EI toward the boundary in high dimension and the single GP trained on the entire DoE. At $d = 4$, TURBO remains second under multiplicative noise (0.212), whereas under additive noise the other four solvers lie within 0.04 of each other. At $d = 50$, NSMA-BO becomes second under both types of noise (0.383 and 0.358), and TREGO remains last (0.584 and 0.454). TURBO falls to third under additive noise, level with BAXUS (0.458 and 0.463), and to fourth under multiplicative noise (0.436).



Normalized final objective value

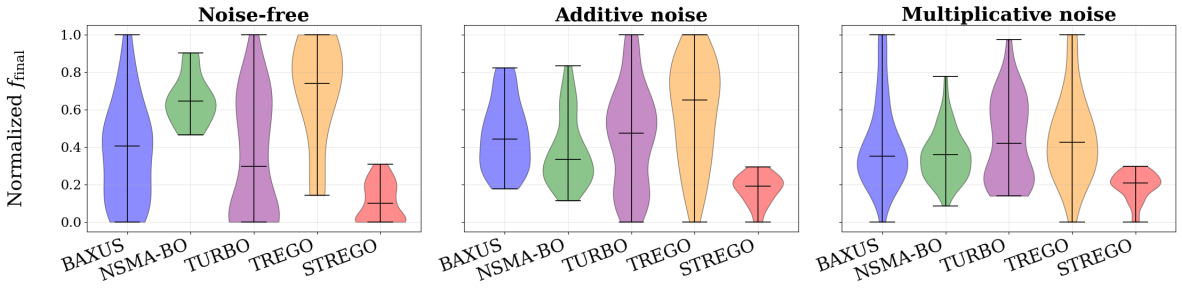
Solver	Noise-free	Additive	Multiplicative
BAXUS	0.202 ± 0.032	0.511 ± 0.054	0.301 ± 0.044
NSMA-BO	0.171 ± 0.026	0.494 ± 0.048	0.436 ± 0.041
TURBO	0.136 ± 0.026	0.474 ± 0.050	0.212 ± 0.034
TREGO	0.165 ± 0.038	0.499 ± 0.045	0.252 ± 0.047
STREGO	$\mathbf{0.080} \pm 0.013$	$\mathbf{0.149} \pm 0.023$	$\mathbf{0.111} \pm 0.016$

CPU time (s)

Solver	Noise-free	Additive	Multiplicative
BAXUS	76.98 ± 2.06	64.30 ± 5.28	60.76 ± 2.83
NSMA-BO	452.45 ± 14.37	409.16 ± 15.38	389.02 ± 11.89
TURBO	$\mathbf{13.50} \pm 0.36$	$\mathbf{11.94} \pm 0.62$	$\mathbf{13.15} \pm 0.57$
TREGO	35.00 ± 1.42	29.88 ± 1.45	27.89 ± 1.09
STREGO	347.29 ± 10.56	298.41 ± 27.44	279.63 ± 9.73

Figure 5: Normalized final objective value (lower is better) and CPU time across different noise settings for $d = 4$. Tables report the mean $\pm$ standard error over 20 trials and 4 functions.

In terms of cost, STREGO is expensive for $d \leq 20$ but not for $d \geq 50$. At $d = 4$, it is the second most expensive solver (280 to 347 s), after NSMA-BO (389 to 452 s), whereas TURBO, TREGO and BAXUS need 12 to 77 s. A plausible cause is that both STREGO and NSMA-BO solve a bi-objective



Normalized final objective value

Solver	Noise-free	Additive	Multiplicative
BAXUS	0.376 ± 0.038	0.463 ± 0.025	0.419 ± 0.031
NSMA-BO	0.652 ± 0.041	0.383 ± 0.026	0.358 ± 0.018
TURBO	0.339 ± 0.046	0.458 ± 0.037	0.436 ± 0.030
TREGO	0.694 ± 0.040	0.584 ± 0.037	0.454 ± 0.031
STREGO	$\mathbf{0.107} \pm 0.015$	$\mathbf{0.178} \pm 0.010$	$\mathbf{0.195} \pm 0.010$

CPU time (s)

Solver	Noise-free	Additive	Multiplicative
BAXUS	609.87 ± 27.65	615.45 ± 38.17	591.74 ± 32.77
NSMA-BO	$\mathbf{372.56} \pm 11.50$	473.16 ± 39.20	494.36 ± 31.38
TURBO	515.66 ± 24.66	550.57 ± 24.11	561.89 ± 23.05
TREGO	913.53 ± 9.25	1382.85 ± 85.35	1270.90 ± 74.19
STREGO	449.19 ± 32.99	$\mathbf{356.40} \pm 21.30$	$\mathbf{384.79} \pm 23.96$

Figure 6: Normalized final objective value (lower is better) and CPU time across different noise settings for $d = 50$. Tables report the mean $\pm$ standard error over 20 trials and 5 functions.

problem with NSMA at every global iteration, which costs more than the single-objective acquisition step of the other solvers. As the dimension grows, the cost of STREGO increases moderately, from 334 s at $d = 20$ to 519 s at $d = 100$, whereas that of TREGO rises from 177 s to 6272 s. For $d \geq 50$ in the noise-free setting, only NSMA-BO is cheaper than STREGO (373 and 503 s, as opposed to 449 and 519 s). Under noise at $d = 50$, STREGO becomes the cheapest solver (356 and 385 s), while the cost of TREGO increases to 1383 and 1271 s.

5.5.2 Results on real-world and engineering benchmarks

The six real-world problems are Rover (60d), Mopta08 (124d) and four MuJoCo locomotion tasks: Swimmer (16d), Hopper (33d), Walker (102d) and HalfCheetah (102d). We add no artificial noise to them. Rover and Mopta08 are deterministic, whereas the MuJoCo tasks are inherently noisy, because each evaluation returns the cumulative reward of a policy whose initial state is randomly perturbed by the simulator. Since STREGO does not handle the blackbox constraints of Mopta08, we add them to the objective as penalty terms, as is common in high-dimensional BO [19, 38]. Figure 7 shows the evolution of f_{best}, the only metric reported for these problems.

STREGO outperforms all other solvers on Walker and Mopta08, where it takes the lead early and ends at ≈ -800 and ≈ 262, respectively, as opposed to ≈ -490 for BAXUS and ≈ 275 for TURBO, the next best solvers. On HalfCheetah, it ends slightly lower than BAXUS and far ahead of the other solvers. On Swimmer, all solvers end close to each other, with STREGO and TREGO slightly ahead. The remaining two problems favor other solvers. On Hopper, TURBO and BAXUS end lower than STREGO (≈ -1700 and ≈ -1300, as opposed to ≈ -1100), and on Rover, TREGO improves much faster at first and ends slightly lower (≈ 6.3 as opposed to ≈ 6.5).

The relative performance of the other four solvers varies from one problem to another, which suggests that the problems differ in structure. The good results of BAXUS on Walker and HalfCheetah point to an exploitable low-dimensional structure in these problems. BAXUS is weaker on Swimmer and Rover, whose active subspace may be too high-dimensional for its random embeddings to capture within the budget. On Mopta08, which is known to have a high effective dimension [38], it ranks only third. TREGO ranks third on HalfCheetah, fourth on Mopta08, second to last on Walker and last on Hopper, where it barely improves after about 100 evaluations. A plausible cause is that TREGO trains a single GP on the entire DoE, whose lengthscales may not fit the neighborhood of the incumbent once the trust region shrinks (Section 4.2). NSMA-BO stalls early on Walker and Mopta08, and its progress

on Mopta08 is essentially flat after about 50 evaluations. This behavior may come from its variance objective, which favors isolated points near the boundary, where little is learned in high dimension (Section 4.1). STREGO is designed to avoid both limitations, with IVR in the global phase and a local GP in the local phase, and it imposes no restriction on the effective dimension. We believe that these properties explain, at least in part, why it performs well both on Walker and HalfCheetah, where BAXUS is also strong, and on Mopta08, which has a high effective dimension.

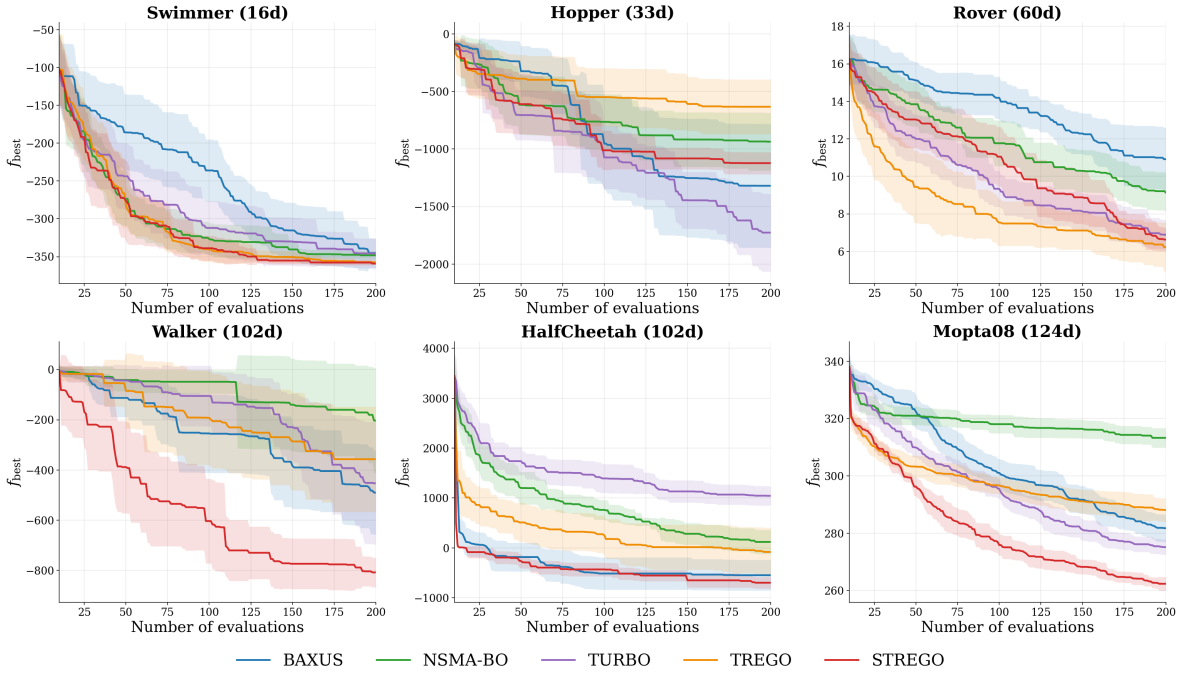


Figure 7: Convergence of STREGO, TREGO, TURBO, BAXUS and NSMA-BO on six real-world benchmarks of increasing dimensionality (16d to 124d).

6 Conclusion

At the outset, we asked whether a trust-region BO method could keep its convergence guarantees with noisy evaluations and remain competitive in high dimension. Our results indicate that it can. On the theoretical side, our framework extends the convergence analysis of TREGO to noisy function estimates that are sufficiently accurate with sufficiently high probability. On the practical side, the performance of STREGO seems largely due to two design choices. Its global phase ranks candidates by the uncertainty they remove over the whole domain rather than by their own uncertainty, and its local phase fits its model only to the points near the incumbent.

In our experiments, replacing the predictive variance by the integrated variance reduction generally improved the final values, except on Schwefel. STREGO gave the best solutions on average in every synthetic configuration tested, from dimension 4 to 100, with and without noise, whereas TREGO ranked last from dimension 50 on. On the real-world problems, STREGO found the best solution on Walker, Mopta08 and HalfCheetah and tied with TREGO on Swimmer, while TURBO and BAXUS did better on Hopper and TREGO on Rover. Unlike BAXUS, which searches low-dimensional subspaces, STREGO searches the full space, and it performed well both on Walker and HalfCheetah, where BAXUS is strong, and on Mopta08, where BAXUS ranks third. Its weakness is its computational cost at low dimension. Up to dimension 20, it was more expensive than three of the four other solvers, whereas from dimension 50 on, only NSMA-BO was cheaper, and only without noise.

Future research will be devoted to three extensions of STREGO. The first is a cheaper way to solve the bi-objective problem of the global phase, to reduce the cost of STREGO at low dimension. The second is the handling of categorical and integer variables, which are common in the tuning of machine learning models, a setting where evaluations are noisy. The third is the direct handling of blackbox constraints and of simulations of varying fidelity, both common in aircraft design [41]. Because the framework leaves the estimation procedure open, we will also examine whether low-fidelity simulations can provide sufficiently accurate estimates.

A Synthetic test problems

COCO/BBOB-inspired synthetic functions. Our first benchmark set is inspired by the COmparing Continuous Optimizers (COCO) platform [23] and in particular its BBOB test suite. We consider five standard analytical functions: Ackley, Alpine-01, Rotated Rastrigin, Schwefel and Hartmann-6. For each function, we report results in dimensions $d \in \{4, 20, 50, 100\}$ except for Hartmann-6 which only allows $d \geq 6$.

Ackley function. The Ackley function is a widely used multimodal benchmark, characterized by a nearly flat outer region and a deep, narrow basin around the global optimum.

$$f_{\text{Ackley}}(x) = -20 \exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right) + 20 + e.$$

The global minimum is $f(x^\star) = 0$ at $x^\star = (0, \ldots, 0)$, for $x \in [-32.768, 32.768]^d$.

Alpine function (Alpine-01). The Alpine-01 function is a nonsmooth benchmark with sharp ridges, commonly used to assess robustness to nondifferentiability.

$$f_{\text{Alpine}}(x) = \sum_{i=1}^{d} \left| x_i \sin(x_i) + 0.1 x_i \right|.$$

The global minimum is $f(x^\star) = 0$ at $x^\star = (0, \ldots, 0)$, for $x \in [-10, 10]^d$.

Schwefel function. The Schwefel function is a benchmark whose global optimum lies geometrically distant from the next-best local minima.

$$f_{\text{Schwefel}}(x) = 418.9829\, d - \sum_{i=1}^{d} x_i \sin\left(\sqrt{|x_i|}\right).$$

The global minimum is $f(x^\star) = 0$ at $x^\star = (420.9687, \ldots, 420.9687)$, for $x \in [-500, 500]^d$.

Hartmann-6 function. The six-dimensional Hartmann function is a multimodal benchmark with six local minima and a narrow basin around the global optimum.

$$f_{\text{Hartmann6}}(x) = -\sum_{i=1}^{4} \alpha_i \exp\left(-\sum_{j=1}^{6} A_{ij}\left(x_j - P_{ij}\right)^2\right),$$

where $\alpha = (1.0,\ 1.2,\ 3.0,\ 3.2)^\top$,

$$A = \begin{pmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{pmatrix}, \qquad P = 10^{-4} \begin{pmatrix} 1312 & 1696 & 5569 & 124 & 8283 & 5886 \\ 2329 & 4135 & 8307 & 3736 & 1004 & 9991 \\ 2348 & 1451 & 3522 & 2883 & 3047 & 6650 \\ 4047 & 8828 & 8732 & 5743 & 1091 & 381 \end{pmatrix}.$$

The global minimum is $f(x^\star) \approx -3.32237$ at $x^\star \approx (0.20169, 0.15001, 0.47687, 0.27533, 0.31165, 0.65730)$, for $x \in [0,1]^6$. We embed the standard 6-dimensional Hartmann function in higher dimensions ($d > 6$) by evaluating it on six coordinates chosen at random (fixed for each d) and leaving the other coordinates as inactive dummy variables that do not affect the objective.

Rotated Rastrigin function. The rotated Rastrigin function is a highly multimodal benchmark with a regular grid of local minima. An orthogonal rotation couples the coordinates, so the problem cannot be split into one-dimensional subproblems.

$$f_{\text{RotRastrigin}}(x) = 10\,d + \sum_{i=1}^{d} \left[z_i^2 - 10\cos(2\pi z_i)\right], \qquad z = R\,x,$$

where $R \in \mathbb{R}^{d\times d}$ is a fixed random orthogonal matrix. The global minimum is $f(x^\star) = 0$ at $x^\star = (0,\ldots,0)$, for $x \in [-5.12, 5.12]^d$.

Noisy variants. To evaluate robustness under stochastic evaluations, we construct noisy variants of each synthetic function using two noise settings commonly used in the noisy blackbox optimization literature [50].

Additive Gaussian noise corrupts the objective with noise that is calibrated to each function's output scale:

$$\tilde{f}(x) = f(x) + \sigma_{f,d}\epsilon, \epsilon \sim \mathcal{N}(0,1),$$

where $\sigma_{f,d}$ is the function and dimension specific noise standard deviation, calibrated to approximately 10% of the output range for function f in dimension d. We estimate the output range of each objective via Monte Carlo sampling. For each (f,d) pair, we uniformly sample 10,000 random points from the domain, evaluate f at each point and compute the observed range $R_{f,d} = \max_i f(x_i) - \min_i f(x_i)$. We then set $\sigma_{f,d} = 0.10 \times R_{f,d}$.

Multiplicative Gaussian noise instead scales the perturbation with the function value:

$$\tilde{f}(x) = f(x)\big(1 + \sigma\epsilon\big), \epsilon \sim \mathcal{N}(0,1),$$

where $\sigma = 0.1$.

Shifting. Ackley, Alpine-01 and Rotated Rastrigin have one of their global optima at the origin $x^\star = (0,\ldots,0)$, the exact center of the search domain. Evaluating them unshifted would favor solvers whose sampling is biased toward the center. Following the idea of [51], we therefore evaluate shifted variants in which every coordinate is displaced:

$$f_{\text{shifted}}(x) = f(x - s), \qquad s_i \sim \mathcal{U}\big(x_{\text{LB}} + 0.2\,\Delta,\ x_{\text{UB}} - 0.2\,\Delta\big), \quad i = 1,\ldots,d,$$

where $\Delta = x_{\text{UB}} - x_{\text{LB}}$ is the width of the domain. The shifted optimum $x^\star = s$ thus lies uniformly in the central 60% of the domain along each coordinate. All three functions are nonnegative on $\mathbb{R}^d$ and attain their minimum value 0 at the origin, so the region of $x - s$ that falls outside the canonical domain cannot contain a better minimum.

Schwefel has the opposite problem. Its optimum $x_i^\star = 420.9687$ lies at 92.1% of the domain $[-500, 500]$ in every coordinate, close to the upper bound. This placement favors solvers biased toward the boundary and penalizes solvers whose acquisition criteria integrate over the domain. Unlike the other three functions, Schwefel is not bounded below outside its canonical domain, so a large shift would let $x - s$ reach regions whose values compete with, or fall below, the true optimum. We therefore draw a negative shift from a narrow range:

$$f_{\text{shifted}}(x) = f(x - s), \qquad s_i \sim \mathcal{U}(-140, -125), \quad i = 1,\ldots,d.$$

This moves the optimum inward to $x_i^\star = 420.9687 + s_i \in [281.0, 296.0]$, that is, to 78–80% of the domain, while keeping $f(x^\star) = 0$. The shift is capped at 140 in magnitude so that the region of $x_i - s_i$ outside the canonical domain, $(500, 640]$, contributes a per-coordinate value of at least 313.6. That is well above the second-best local minimum inside the original domain (118.4). The global optimum therefore stays unique. Moreover, since $x_i - s_i \in [-375, 640]$, all seven per-coordinate local minima of the original function remain inside the search domain, so the function is just as multimodal as before.

For every function, the shift vector s is drawn once for each dimension d from a fixed seed and the same s is used in all trials and for all solvers.

B Hyperparameters

All experiments use the same STREGO configuration across benchmarks, dimensions and noise regimes. The settings are summarized in Table B1.

Table B1: STREGO hyperparameters used for a d-dimensional problem.

Hyperparameter	Setting
Initial DoE size (N_0)	10
Evaluation budget	200
Number of trials	20
Base seed	42
Initial trust-region radius (δ_0)	$0.5\,(1/5)^{1/d}$
Expansion factor (γ)	2.0
Contraction factor, empty iteration (β_0)	0.7
Contraction factor, uncertain unsuccessful (β_1)	0.7
Contraction factor, certain unsuccessful (β_2)	0.5
κ	1.0
Minimum number of local points ($n_{\min}$)	10
Global batch size (G)	3
$r_{\max}$	1
$r_{\min}$	$\epsilon_m \approx 2.2 \times 10^{-16}$

C Sensitivity analysis: Global batch size

Figure C1 compares STREGO with the global batch sizes $\mathrm{G} \in \{1, 2, 3, 4\}$, with all other settings and the evaluation budget held fixed. At $d = 4$, $\mathrm{G} = 1$ performs best on all four functions and in all three noise settings, except on Schwefel under multiplicative noise, where the confidence bands of all variants overlap. For $d \geq 20$, the variants with $\mathrm{G} > 1$ generally match or outperform $\mathrm{G} = 1$ on Ackley, Alpine-01 and Rotated Rastrigin. The gap is clearest on Ackley under additive noise, where, at $d = 50$, the variants with $\mathrm{G} > 1$ end below 18, as opposed to about 19 for $\mathrm{G} = 1$. Schwefel is again the exception, as $\mathrm{G} = 1$ performs best and performance generally degrades as G increases.

These results may be explained by the division of the fixed budget between the two phases. A larger batch spends more evaluations on exploring the domain and leaves fewer for the local phase. At $d = 4$, the domain is small enough to be covered with few global evaluations, so additional global candidates mainly reduce the budget of the local phase. Schwefel has many deep local minima and no global shape that guides the search toward the optimum, so progress comes mainly from the local search around the incumbent. In higher dimension, covering the domain is harder, and the additional global candidates improve the results. Although $\mathrm{G} = 1$ performs best at $d = 4$, we keep $\mathrm{G} = 3$ in all dimensions because high dimension is our main interest.

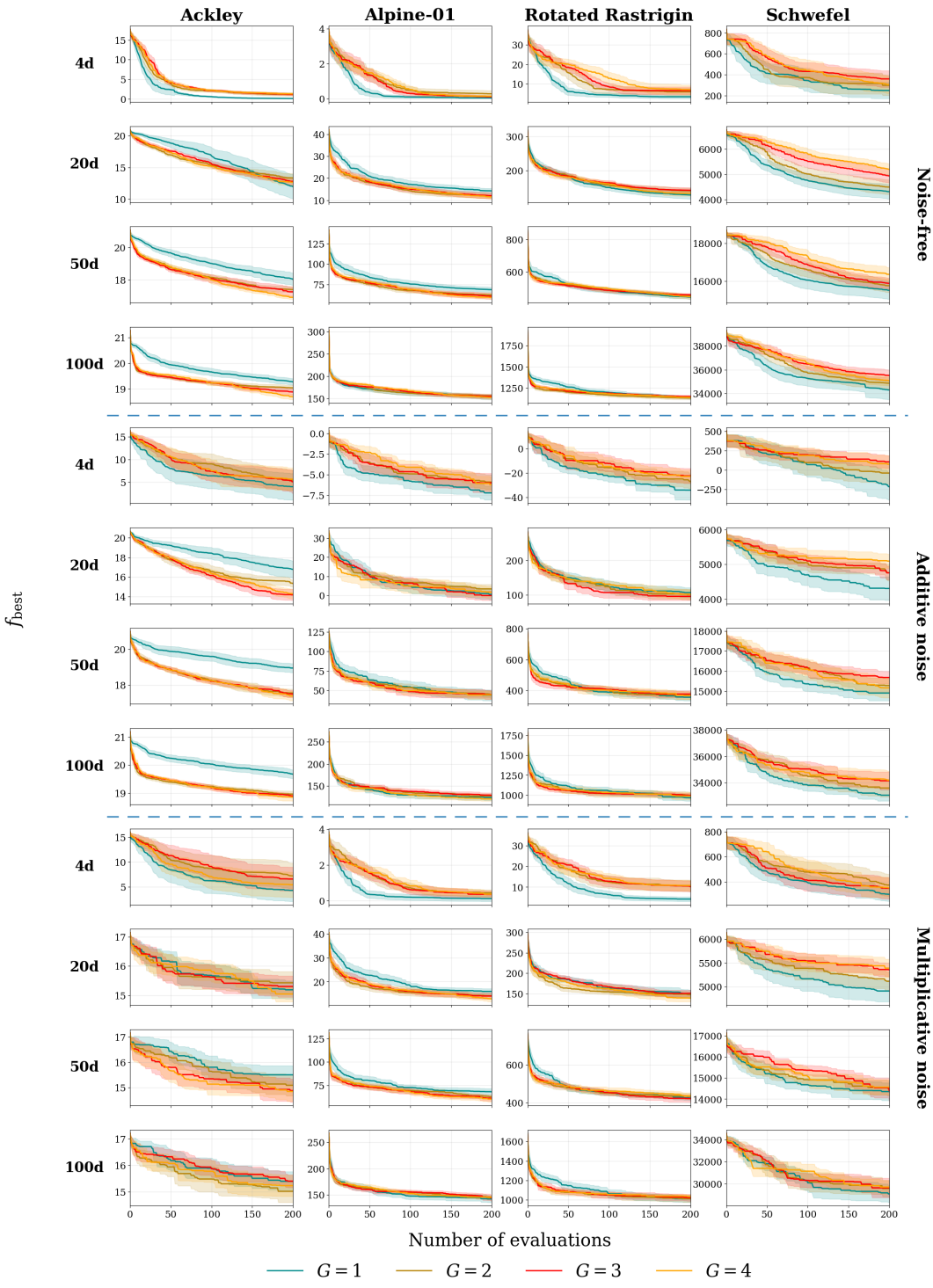


Figure C1: Comparison of STREGO with global batch sizes $\mathrm{G} \in \{1, 2, 3, 4\}$, all other settings held fixed, across dimensions and noise settings, averaged over 20 trials.

References

[1] C. Audet and J. E. Dennis, Jr. Mesh adaptive direct search algorithms for constrained optimization. SIAM Journal on Optimization, 17(1):188–217, 2006.

[2] C. Audet, K. J. Dzahini, M. Kokkolaras, and S. Le Digabel. Stochastic mesh adaptive direct search for blackbox optimization using probabilistic estimates. Computational Optimization and Applications, 79(1):1–34, 2021.

[3] C. Audet and W. Hare. Derivative-Free and Blackbox Optimization. Springer Series in Operations Research and Financial Engineering. Springer, Cham, Switzerland, 2nd edition, 2026. In preparation.

[4] M. Balandat, B. Karrer, D. Jiang, S. Daulton, B. Letham, A. G. Wilson, and E. Bakshy. BoTorch: A framework for efficient Monte-Carlo Bayesian optimization. Advances in Neural Information Processing Systems, 33:21524–21538, 2020.

[5] A. Bardou, P. Thiran, and T. Begin. Relaxing the additivity constraints in decentralized no-regret high-dimensional Bayesian optimization. In Proceedings of the 12th International Conference on Learning Representations (ICLR), 2024.

[6] J. Bect, D. Ginsbourger, L. Li, V. Picheny, and E. Vazquez. Sequential design of computer experiments for the estimation of a probability of failure. Statistics and Computing, 22(3):773–793, 2012.

[7] M. Binois, A. Fadikar, and A. Stevens. Gearing Gaussian process modeling and sequential design towards stochastic simulators. arXiv preprint arXiv:2412.07306, 2024.

[8] B. Bischl, S. Wessing, N. Bauer, K. Friedrichs, and C. Weihs. MOI-MBO: Multiobjective infill for parallel model-based optimization. In Learning and Intelligent Optimization (LION 8), volume 8426 of Lecture Notes in Computer Science, pages 173–186. Springer, 2014.

[9] J. Blanchet, C. Cartis, M. Menickelly, and K. Scheinberg. Convergence rate analysis of a stochastic trust-region method via supermartingales. INFORMS Journal on Optimization, 1(2):92–119, 2019.

[10] F. Carciaghi, S. Magistri, P. Mansueto, and F. Schoen. A bi-objective optimization based acquisition strategy for batch Bayesian global optimization. Computational Optimization and Applications, 92(1):81–123, 2025.

[11] R. Chen, M. Menickelly, and K. Scheinberg. Stochastic optimization using a trust-region method and random models. Mathematical Programming, 169(2):447–487, 2018.

[12] S. Daulton, M. Balandat, and E. Bakshy. Parallel Bayesian optimization of multiple noisy objectives with expected hypervolume improvement. Advances in Neural Information Processing Systems, 34:2187–2200, 2021.

[13] G. De Ath, R. M. Everson, A. A. M. Rahat, and J. E. Fieldsend. Greed is good: Exploration and exploitation trade-offs in Bayesian optimisation. ACM Transactions on Evolutionary Learning and Optimization, 1(1):1–22, 2021.

[14] A. Ding, T. H. Tran, and L. N. Vicente. Non-monotone direct-search methods for deterministic and stochastic derivative-free optimization. arXiv preprint arXiv:2609.11567, 2026.

[15] Y. Diouane, A. Lucchi, and V. Patil. A globally convergent evolutionary strategy for stochastic constrained optimization with applications to reinforcement learning. In Proceedings of the 25th International Conference on Artificial Intelligence and Statistics, volume 151 of PMLR, pages 836–859, 2022.

[16] Y. Diouane, V. Picheny, R. Le Riche, and A. Scotto Di Perrotolo. TREGO: A trust-region framework for efficient global optimization. Journal of Global Optimization, 86(1):1–23, 2023.

[17] R. Durrett. Probability: Theory and Examples, volume 49 of Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 5th edition, 2019.

[18] D. Eriksson and M. Jankowiak. High-dimensional Bayesian optimization with sparse axis-aligned subspaces. In Proceedings of the 37th Conference on Uncertainty in Artificial Intelligence (UAI), volume 161, pages 493–503. PMLR, 2021.

[19] D. Eriksson, M. Pearce, J. Gardner, R. D. Turner, and M. Poloczek. Scalable global optimization via local Bayesian optimization. Advances in Neural Information Processing Systems, 32:5496–5507, 2019.

[20] R. Garnett. Bayesian Optimization. Cambridge University Press, 2023.

[21] D. Ginsbourger, R. Le Riche, and L. Carraro. Kriging is well-suited to parallelize optimization. In Computational Intelligence in Expensive Optimization Problems, volume 2 of Adaptation, Learning, and Optimization, pages 131–162. Springer, 2010.

[22] J. Guo, B. Ranković, and P. Schwaller. Bayesian optimization for chemical reactions. Chimia, 77(1/2):31–38, 2023.

[23] N. Hansen, A. Auger, R. Ros, O. Mersmann, T. Tušar, and D. Brockhoff. COCO: A platform for comparing continuous optimizers in a black-box setting. Optimization Methods and Software, 36(1):114–144, 2021.

[24] C. Hvarfner, E. O. Hellsten, and L. Nardi. Vanilla Bayesian optimization performs great in high dimensions. In Proceedings of the 41st International Conference on Machine Learning (ICML), volume 235, pages 20793–20817. PMLR, 2024.

[25] C. Jiang and M. Li. Trading off quality and uncertainty through multi-objective optimisation in batch Bayesian optimisation. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 39, pages 27027–27035, 2025.

[26] D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. Journal of Global Optimization, 13(4):455–492, 1998.

[27] K. Kandasamy, J. Schneider, and B. Póczos. High dimensional Bayesian optimisation and bandits via additive models. In Proceedings of the 32nd International Conference on Machine Learning (ICML), volume 37, pages 295–304. PMLR, 2015.

[28] R. Lafage. egobox, a Rust toolbox for efficient global optimization. Journal of Open Source Software, 7(78):4737, 2022.

[29] M. Lapucci, P. Mansueto, and F. Schoen. A memetic procedure for global multi-objective optimization. Mathematical Programming Computation, 15(2):227–267, 2023.

[30] S. Le Digabel and S. M. Wild. A taxonomy of constraints in black-box simulation-based optimization. Optimization and Engineering, 25(2):1125–1143, 2024.

[31] W. Lyu, F. Yang, C. Yan, D. Zhou, and X. Zeng. Batch Bayesian optimization via multi-objective acquisition ensemble for automated analog circuit design. In Proceedings of the 35th International Conference on Machine Learning (ICML), volume 80, pages 3306–3314. PMLR, 2018.

[32] J. Mockus. On Bayesian methods for seeking the extremum. In Optimization Techniques IFIP Technical Conference, Novosibirsk, July 1–7, 1974, volume 27 of Lecture Notes in Computer Science, pages 400–404. Springer, 1975.

[33] J. Mockus. The Bayesian approach to local optimization. In Bayesian Approach to Global Optimization: Theory and Applications, pages 125–156. Kluwer Academic Publishers, 1989.

[34] R. Moriconi, M. P. Deisenroth, and K. S. Sesh Kumar. High-dimensional Bayesian optimization using low-dimensional feature spaces. Machine Learning, 109(9):1925–1943, 2020.

[35] N. Namura and S. Takemori. Regional expected improvement for efficient trust region selection in high-dimensional Bayesian optimization. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 39, pages 19624–19632, 2025.

[36] C. Oh, E. Gavves, and M. Welling. BOCK: Bayesian optimization with cylindrical kernels. In Proceedings of the 35th International Conference on Machine Learning (ICML), volume 80, pages 3868–3877. PMLR, 2018.

[37] L. Papenmeier and L. Nardi. Bencher: Simple and reproducible benchmarking for black-box optimization. In ICML 2025 Workshop on Championing Open-source Development in Machine Learning (CODEML), 2025. arXiv:2505.21321.

[38] L. Papenmeier, L. Nardi, and M. Poloczek. Increasing the scope as you learn: Adaptive Bayesian optimization in nested subspaces. Advances in Neural Information Processing Systems, 35:11586–11601, 2022.

[39] C. Paquette and K. Scheinberg. A stochastic line search method with expected complexity analysis. SIAM Journal on Optimization, 30(1):349–376, 2020.

[40] V. Picheny. A Stepwise uncertainty reduction approach to constrained global optimization. In Proceedings of the 17th International Conference on Artificial Intelligence and Statistics (AISTATS), volume 33, pages 787–795. PMLR, 2014.

[41] R. Priem, H. Gagnon, I. Chittick, S. Dufresne, Y. Diouane, and N. Bartoli. An efficient application of Bayesian optimization to an industrial MDO framework for aircraft design. In AIAA AVIATION 2020 Forum, page 3152, 2020.

[42] C. E. Rasmussen and C. K. I. Williams. Gaussian Processes for Machine Learning. MIT Press, 2006.

[43] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn. Design and analysis of computer experiments. Statistical Science, 4(4):409–423, 1989.

[44] P. Saves, R. Lafage, N. Bartoli, Y. Diouane, J. Bussemaker, T. Lefebvre, J. T. Hwang, J. Morlier, and J. R. R. A. Martins. SMT 2.0: A surrogate modeling toolbox with a focus on hierarchical and mixed variables Gaussian processes. Advances in Engineering Software, 188:103571, 2024.

[45] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas. Taking the human out of the loop: A review of Bayesian optimization. Proceedings of the IEEE, 104(1):148–175, 2016.

[46] B. J. Shields, J. Stevens, J. Li, M. Parasram, F. Damani, J. I. Martinez Alvarado, J. M. Janey, R. P. Adams, and A. G. Doyle. Bayesian reaction optimization as a tool for chemical synthesis. Nature, 590(7844):89–96, 2021.

[47] E. Siivola, A. Vehtari, J. Vanhatalo, J. González, and M. R. Andersen. Correcting boundary over-exploration deficiencies in Bayesian optimization with virtual derivative sign observations. In 2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP), pages 1–6. IEEE, 2018.

[48] J. Snoek, H. Larochelle, and R. P. Adams. Practical Bayesian optimization of machine learning algorithms. Advances in Neural Information Processing Systems, 25:2951–2959, 2012.

[49] W.-T. Tang and J. A. Paulson. Rethinking trust region Bayesian optimization in high dimensions. In AISTATS 2026 Workshop OPTIMAL, 2026. arXiv:2604.22967.

[50] K. Uchida, K. Nishihara, and S. Shirakawa. CMA-ES with adaptive reevaluation for multiplicative noise. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), pages 731–739, 2024.

[51] R. T.-Y. Yu, C. Picard, and F. Ahmed. GIT-BO: High-dimensional Bayesian optimization with tabular foundation models. In International Conference on Learning Representations (ICLR), 2026.

[52] S. Zhang, Q. Yang, R. Wang, and X. Wang. Multi-objective Bayesian optimization design of elliptical double serpentine nozzle. Aerospace, 11(1):48, 2024.

[53] A. Žilinskas and J. Calvin. Bi-objective decision making in global optimization based on statistical models. Journal of Global Optimization, 74(4):599–609, 2019.

[54] J. K. Ziomek and H. Bou Ammar. Are random decompositions all we need in high dimensional Bayesian optimisation? In Proceedings of the 40th International Conference on Machine Learning (ICML), volume 202, pages 43347–43368. PMLR, 2023.