Every Call is Precious: Global Optimization of Black-Box Functions with Unknown Lipschitz Constants

Fares Fourati KAUST
Salma Kharrat KAUST
Vaneet Aggarwal Purdue University
Mohamed-Slim Alouini KAUST

Abstract

Optimizing expensive, non-convex, black-box Lipschitz continuous functions presents significant challenges, particularly when the Lipschitz constant of the underlying function is unknown. Such problems often demand numerous function evaluations to approximate the global optimum, which can be prohibitive in terms of time, energy, or resources. In this work, we introduce *Every Call is Precious (ECP)*, a novel global optimization algorithm that minimizes unpromising evaluations by strategically focusing on potentially optimal regions. Unlike previous approaches, ECP eliminates the need to estimate the Lipschitz constant, thereby avoiding additional function evaluations. ECP guarantees no-regret performance for infinite evaluation budgets and achieves minimax-optimal regret bounds within finite budgets. Extensive ablation studies validate the algorithm's robustness, while empirical evaluations show that ECP outperforms 10 benchmark algorithms—including Lipschitz, Bayesian, bandits, and evolutionary methods—across 30 multi-dimensional non-convex synthetic and real-world optimization problems, which positions ECP as a competitive approach for global optimization.

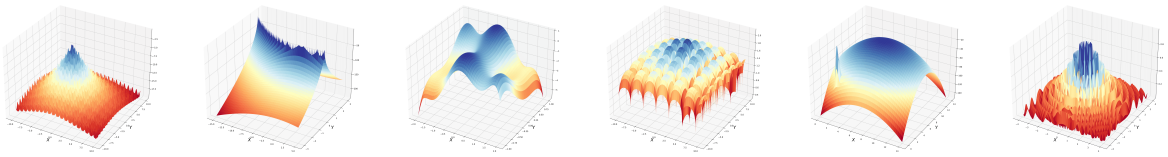
1 INTRODUCTION

Global optimization is an evolving field of optimization (Törn and Žilinskas, 1989; Pardalos, 2013; Floudas and Pardalos, 2014; Zabinsky, 2013; Stork et al., 2022) that seeks to identify the best possible solution across the entire problem space, i.e., the entire set of feasible solutions, ensuring that the global optimum is found or at least well-approximated, even with a limited number of function evaluations. The objective function may be non-convex and exhibit multiple local optima. As a result, local optimization methods may only find a solution that is optimal in a limited region of the search space, potentially yielding a globally suboptimal result. Moreover, the objective function can be non-differentiable or a black-box function (Jones et al., 1998), meaning it is only accessible through direct evaluations. Furthermore, evaluating the objective function can be expensive, requiring substantial amounts of money, time, or energy, which makes the process even more challenging, as it can limit the number of function evaluations.

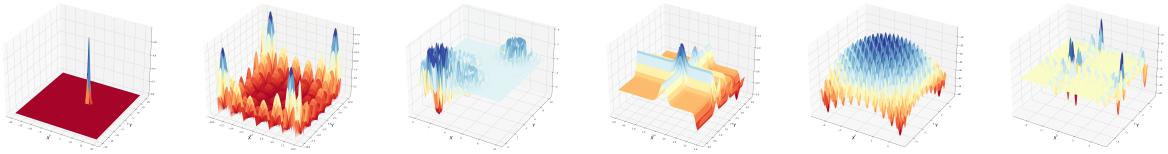
Despite the challenges, global optimization problems are prevalent in engineering and real-world systems. These problems are applicable in various fields, including mechanical, civil, and chemical engineering, as well as in structural optimization, molecular biology, circuit chip design, and image processing (Zabinsky, 2013). More recently, with the emergence of large language models (LLMs), a new line of work has focused on instruction learning in black-box LLMs, such as ChatGPT (OpenAI, 2023), through calls, without access to their underlying models (Chen et al., 2024; Lin et al., 2024; Kharrat et al., 2024). Therefore, the development of efficient global optimization algorithms represents an intriguing research direction with the potential for significant impact across several disciplines.

A subfield of global optimization is Lipschitz optimization (Shubert, 1972; Piyavskii, 1972), which assumes knowledge of the Lipschitz constant or an upper bound for it. While Törn and Žilinskas (1989) observed that practical objective functions often exhibit Lipschitz continuity, the exact value of this constant is seldom known. In this work, we focus on black-box Lipschitz continuous functions with *unknown* constants. In such cases, a common approach is to estimate the Lipschitz constant or an upper bound and use this estimate as a proxy for the true constant (Mal-

Proceedings of the 28[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258.



(a) Ackley (b) Bukin (c) Camel (d) Crossintray (e) Damavandi (f) Dropwave



(g) Easom (h) Holder (i) Langermann (j) Michalewicz (k) Rastrigin (l) Schubert

Figure 1: Selected figures of various considered non-convex objective functions with two dimensions.

herbe and Vayatis, 2017; Serré et al., 2024). Unlike these previous works, our approach bypass Lipschitz constant estimation—typically reliant on random uniform evaluations—and concentrate solely on evaluating points that are potential maximizers. This significantly improves the efficiency of the global optimization process, particularly for expensive objective functions with tight evaluation budgets.

We introduce *Every Call is Precious* (ECP) a simple yet efficient approach that avoids unpromising uniform random evaluations. The core idea is to start with small acceptance regions, possibly empty, controlled by a variable, ε_t. This region leverages the Lipschitz continuity of the function—without requiring the Lipschitz constant—by using ε_t and the points observed in previous iterations. The proposed region is theoretically guaranteed, for smaller values of t, to be a subset of potential maximizers, ensuring that every evaluated point is a viable candidate for the maximum. ECP gradually expands the acceptance region through a growing sequence of ε_t, progressively including more potential maximizers until all are covered when $\varepsilon_t \geq k$. This method is especially practical for small evaluation budgets, as it ensures that all evaluated points are potential maximizers, avoiding unpromising evaluations.

In the following, we summarize the contributions:

(1) We introduce ECP, a global optimization algorithm that eliminates the need to estimate a Lipschitz constant. Instead, we propose a simple yet effective adaptive search strategy based on a growing sequence of ε_t. The implementation is publicly available at https://github.com/fouratifares/ECP.

(2) We provide theoretical analysis of ECP, addressing its unique interplay between acceptance region expansion (Lemma 1) and contraction over time (Proposition 3). Key results include finite computational complexity (Theorem 1), no-regret guarantees in the infinite-budget setting (Theorem 2), and minimax optimality with finite budgets (Theorem 3).

(3) Benchmarks against 10 global optimization methods on 30 non-convex, multi-dimensional problems demonstrate that ECP consistently outperforms state-of-the-art Lipschitz, Bayesian, bandits, and evolutionary methods. Furthermore, extensive experiments validate ECP's robustness to hyperparameter choices.

2 RELATED WORKS

Several methods have been proposed for global optimization, with the simplest being non-adaptive exhaustive searches, such as grid search, which uniformly divides the space into representative points (Zabinsky, 2013), or its stochastic alternative, Pure Random Search (PRS), which employs random uniform sampling (Brooks, 1958; Zabinsky, 2013). However, these methods are often inefficient, as they fail to exploit previously gathered information or the underlying structure of the objective function (Zabinsky, 2013).

To enhance efficiency, adaptive methods have been developed that leverage collected data and local smoothness. Some of these methods need the knowledge of the local smoothness, including HOO (Bubeck et al., 2011), Zooming (Kleinberg et al., 2008), and DOO (Munos, 2011), while others do not, such as SOO (Munos, 2011; Preux et al., 2014; Kawaguchi et al., 2016) and SequOOL (Bartlett et al., 2019). In this work, however, we focus on Lipschitz functions.

To address Lipschitz functions with unknown Lipschitz constants, the DIRECT algorithm (Jones et al., 1993; Jones and Martins, 2021) employs a deterministic splitting approach of the whole space, sequentially dividing and evaluating the function over subdivisions that have recorded the highest upper bounds.

More recently, Malherbe and Vayatis (2017) introduced AdaLIPO, an adaptive stochastic no-regret strategy that estimates the Lipschitz constant through uniform random sampling, which is then used to identify potentially optimal maximizers based on previously explored points. Later, AdaLIPO+ (Serré et al., 2024) was introduced as an empirical enhancement over it, reducing the exploration probability over time. Both approaches optimize the search space using an acceptance condition, yet they necessitate additional uniform random evaluations, making them less efficient in small-budget scenarios.

Under alternative assumptions, various global optimization methods have been proposed. For instance, Bayesian optimization (BO) (Nogueira, 2014–; Shahriari et al., 2016; Frazier, 2018; Balandat et al., 2020) constructs a probabilistic model of the objective function and uses it to evaluate the most promising points, making it particularly effective for global optimization.

While several BO algorithms are theoretically guaranteed to converge to the global optimum of the unknown function, they often rely on the assumption that the kernel's hyperparameters are known in advance. To address this limitation, hyperparameter-free approaches such as Adaptive GP-UCB (A-GP-UCB) (Berkenkamp et al., 2019) have been proposed. More recently, Lindauer et al. (2022) introduced SMAC3 as a robust baseline for global optimization. In our empirical evaluation, we show that ECP outperforms these recent baselines from BO.

Other approaches, such as CMA-ES (Hansen and Ostermeier, 1996; Hansen, 2006; Hansen et al., 2019), and simulated annealing (Metropolis et al., 1953; Kirkpatrick et al., 1983), later extended to Dual Annealing (Xiang et al., 1997; Tsallis, 1988; Tsallis and Stariolo, 1996), are also notable, although they do not guarantee no-regret (Malherbe and Vayatis, 2017) or theoretical finite-budget guarantees for Lipschitz functions.

Other related approaches include contextual bandits (Auer, 2002; Langford and Zhang, 2007; Filippi et al., 2010; Valko et al., 2013; Lattimore and Szepesvári, 2020), such as the NeuralUCB algorithm (Zhou et al., 2020), which leverages neural networks to estimate upper-confidence bounds. While NeuralUCB is not primarily designed for global maximization, it can be adapted by randomly sampling points, estimating their bounds, evaluating the point with the highest estimate, and retraining the network. However, it may be inefficient for small budgets, as neural networks require a large number of samples to train effectively. Finally, other works on bandits address black-box discrete function maximization (Fourati et al., 2023, 2024c,b), which is not the focus of this work.

3 PROBLEM STATEMENT

In this work, we consider a black-box, non-convex, deterministic, real-valued function f, which may be expensive to evaluate—requiring significant time, energy, or financial resources. The function is defined over a convex, compact set $\mathcal{X} \subset \mathbb{R}^d$ with a non-empty interior and has a maximum over its input space[1].

The objective of this work is global maximization, seeking a global maximizer, defined as follows:

$$x^\star \in \arg\max_{x \in \mathcal{X}} f(x)$$

with a minimal number of function calls. Starting from an initial point x_1 and given its function evaluation $f(x_1)$, adaptive global optimization algorithms leverage past observations to identify potential global optimizers. Specifically, depending on the previous evaluations $(x_1, f(x_1)), \cdots, (x_t, f(x_t))$, it chooses at each iteration $t \geq 1$ a point $x_{t+1} \in \mathcal{X}$ to evaluate and receives its function evaluation $f(x_{t+1})$. After n iterations, the algorithm returns $x_{\hat{\imath}_n}$, one of the evaluated points, where $\hat{\imath}_n \in \arg\max_{i=1,\cdots,n} f(x_i)$, representing the point with the highest evaluation.

To assess the performance of an algorithm $A \in \mathcal{G}$, where $\mathcal{G}$ is the set of global optimization algorithms, over the function f, we consider its regret after n iterations, i.e., after evaluating $x_1, \ldots, x_n$ by A, as follows:

$$\mathcal{R}_{A,f}(n) = \max_{x \in \mathcal{X}} f(x) - \max_{i=1,\cdots,n} f(x_i), \tag{1}$$

measuring the difference between the true maximum and the best evaluation over the n iterations.

We consider f to be Lipschitz with an unknown finite Lipschitz constant k, i.e., there exists an unknown $k \geq 0$, such that for any two points, x and x' in $\mathcal{X}$, the absolute difference between $f(x)$ and $f(x')$ is no more than k times the distance between x and x', i.e.,

$$\forall (x, x') \in \mathcal{X}^2 \quad |f(x) - f(x')| \leq k \cdot \|x - x'\|_2 .$$

Morever, we denote the set of Lipschitz-continuous functions defined on $\mathcal{X}$, with a Lipschitz constant k, as $\text{Lip}(k) := \{f : \mathcal{X} \to \mathbb{R} \text{ s.t. } |f(x) - f(x')| \leq k \cdot \|x - x'\|_2 , \forall (x, x') \in \mathcal{X}^2\}$ and their union $\bigcup_{k \geq 0} \text{Lip}(k)$ denotes the set of all Lipschitz-continuous functions.

We define the notion of no-regret, equivalent to optimization consistency (Malherbe and Vayatis, 2017),

[1]For all $x \in \mathbb{R}^d$, we denote its ℓ_2-norm as $\|x\|_2 = (\sum_{i=1}^d x_i^2)^{\frac{1}{2}}$. We define $B(x, r) = \{x' \in \mathbb{R}^d : \|x - x'\|_2 \leq r\}$ the ball centered in x of radius $r \geq 0$. For any bounded set $\mathcal{X} \subset \mathbb{R}^d$, we define its radius as $\text{rad}(\mathcal{X}) = \max\{r > 0 : \exists x \in \mathcal{X} \text{ where } B(x, r) \subseteq \mathcal{X}\}$ and its diameter as $\text{diam}(\mathcal{X}) = \max_{(x,x') \in \mathcal{X}^2} \|x - x'\|_2$.

where the best evaluation converges to the true maximum in probability, which provides a formal guarantee that the algorithm's regret diminishes with an increasing budget n (number of evaluations).

Definition 1. (No-Regret) *An algorithm $A \in \mathcal{G}$ is no-regret over a set $\mathcal{F}$ of real-valued functions, having a maximum over their domain $\mathcal{X}$, if and only if:*

$$\forall f \in \mathcal{F}, \quad \mathcal{R}_{A,f}(n) \xrightarrow{p} 0,$$

where $\mathcal{R}_{A,f}$, defined in Eq. (1), represents the regret of algorithm A after n evaluations of function f.

In fact, finite-time lower bounds on the minimax regret can be established for the class of functions with a fixed Lipschitz constant k, as we recall below.

Proposition 1. *(Bull (2011))* (Lower Bound) *For any Lipschitz function, $f \in Lip(k)$, with any constant $k \geq 0$ and any $n \in \mathbb{N}^\star$, we have*

$$\inf_{A \in \mathcal{G}} \sup_{f \in \mathrm{Lip}(k)} \mathbb{E}\left[\mathcal{R}_{A,f}(n)\right] \geq c \cdot k \cdot n^{-\frac{1}{d}}$$

where $c = \mathrm{rad}(\mathcal{X}) / (8\sqrt{d})$. The expectation is taken over the n evaluations of f by the algorithm A.

The minimax regret lower bound for optimizing Lipschitz functions shows that the best achievable regret decays as $n^{-\frac{1}{d}}$, where n is the budget of function evaluations and d is the input space dimensionality, which underscores the difficulty of high-dimensional optimization. The regret bounds also scale with the Lipschitz constant k, indicating functions with smaller k are easier to optimize. This result sets performance limits for algorithms, with the best expected regret bounded by $\Theta(k \cdot n^{-\frac{1}{d}})$, which can be recovered by any algorithm with a covering rate of $\mathcal{O}(n^{-1/d})$ (Malherbe and Vayatis, 2017). However, the objective functions used to establish the lower bound of $\Omega(kn^{-1/d})$ often feature spikes that are nearly constant across most of the domain, offering limited practical relevance. Therefore, we focus on developing an algorithm that not only meets theoretical guarantees but also demonstrates outstanding performance across a wide variety of non-convex multi-dimensional objective functions.

4 EVERY CALL IS PRECIOUS

In this section, we present ECP, an efficient algorithm, for maximizing an unknown (possibly expensive) function f without knowing its Lipschitz constant $k \geq 0$. Our proposed algorithm, ECP, presented in Algorithm 1, takes as input the number of function evaluations $n \in \mathbb{N}^\star$ (budget), the search space $\mathcal{X}$, the black-box function f, a value $\varepsilon_1 > 0$, a coefficient $\tau_{n,d} > 1$, and a constant $C > 1$. The algorithm begins by sampling and evaluating a point x_1 uniformly

Algorithm 1 Every Call is Precious (ECP)

Input: $n \in \mathbb{N}^\star$, $\varepsilon_1 > 0$, $\tau_{n,d} > 1$, $C > 1$, $\mathcal{X} \subset \mathbb{R}^d$, f

1: Let $x_1 \sim \mathcal{U}(\mathcal{X})$, Evaluate $f(x_1)$
2: $t \leftarrow 1, \quad h_1 \leftarrow 1, \quad h_2 \leftarrow 0$
3: **while** $t < n$ **do**
4: Let $x_{t+1} \sim \mathcal{U}(\mathcal{X}), \quad h_{t+1} \leftarrow h_{t+1} + 1,$
5: **if** $(h_{t+1} - h_t) > C$ **then** ▷ (Growth Condition)
6: $\varepsilon_t \leftarrow \tau_{n,d} \cdot \varepsilon_t, \quad h_{t+1} \leftarrow 0$
7: **end if**
8: **if** $x_{t+1} \in \mathcal{A}_{\varepsilon_t,t}$ **then** ▷ (Acceptance Condition)
9: Evaluate $f(x_{t+1})$
10: $t \leftarrow t+1, \quad h_t \leftarrow h_{t+1}$
11: $\varepsilon_{t+1} \leftarrow \tau_{n,d} \cdot \varepsilon_t, \quad h_{t+1} \leftarrow 0$
12: **end if**
13: **end while**
14: Return $x_{\hat{i}}$ where $\hat{i} \in \arg\max_{i=1,\cdots,n} f(x_i)$

at random from the entire space $\mathcal{X}$ (line 1). It then proceeds through $n-1$ rounds (each round concludes after one function evaluation), where in each round $t \geq 1$ (up to $t = n-1$), a random variable x_{t+1} is repeatedly sampled uniformly from the input space $\mathcal{X}$ until a sample meets the acceptance condition. Once the condition is satisfied, the sample is evaluated, and the algorithm moves to the next round $t+1$.

ECP accepts (evaluates) the sampled point $x_{t+1} = x$ if and only if the following inequality is verified:

$$\min_{i=1,\cdots,t} \left(f(x_i) + \varepsilon_t \cdot \|x - x_i\|_2\right) \geq \max_{j=1,\cdots,t} f(x_j),$$

where ε_t is a growing sequence staring from ε_1 and continuously multiplied by a coefficient $\tau_{n,d} > 1$. An illustration of the acceptance region for some objective function can be found in Fig. 2, where it can be seen that ε_t controls the acceptance region size. The coefficient $\tau_{n,d} > 1$ is some non-decreasing function of n and d, such as $\tau_{n,d} = \max\{1 + \frac{1}{nd}, \tau\} \geq \tau > 1$.

The algorithm tracks the number of sampled but rejected points during each iteration $t \geq 1$ before increasing ε_t, using the variable h_{t+1}. This variable is initialized to zero (lines 2) and is reset to zero whenever ε_t is increased (line 6 and 11). Additionally, h_t is initialized with the number of rejections from the previous iteration (lines 2 and 10), before acceptance at iteration $t-1$. When the difference between the current and the previous number of rejections exceeds a given threshold $C > 1$ (line 5), ε_t is increased by multiplying it with the factor $\tau_{n,d} > 1$. This growth condition is further analyzed in Proposition 5.

Thus, ε_t grows when a rapidly increasing number of samples is generated without an accepted point (lines 5-6) and also when a sample is evaluated (line 11). While the former type of growth is stochastic and depends on the threshold $C > 1$, the latter is deterministic and occurs regardless of the input parameters. Consequently, it follows that $\varepsilon_t \geq \varepsilon_1 \tau_{n,d}^{t-1}$.

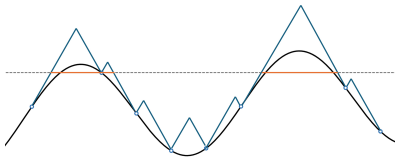


Figure 2: An illustration of the acceptance region (orange), given 8 evaluated points on a single-dimensional objective function (black). The ε_8 controls the slopes of the blue functions, directly impacting the acceptance region.

Formally, for any value of ε_t and at any given time step t, we define $\mathcal{A}_{\varepsilon_t,t}$ as the acceptance region, presenting the set of potentially accepted points at round t with current ε_t. The choice of this condition will be motivated and analyzed in further details in Section 5.1.

Definition 2. (Acceptance Region) *The set of points potentially accepted by ECP at any time step $t \geq 1$, for any value $\varepsilon_t > 0$, is defined as:* $\mathcal{A}_{\varepsilon_t,t} \triangleq \{x \in \mathcal{X} : \min_{i=1,\cdots,t}(f(x_i)+\varepsilon_t \cdot \|x - x_i\|_2) \geq \max_{j=1,\cdots,t} f(x_j)\}$.

This acceptance condition as shown later in Proposition 2, ensures that the points potentially accepted are optimal for smaller values of ε_t and, as ε_t increases, encompass all potentially optimal points. Therefore, unlike PRS, which accepts any sampled point, or other methods that allow for some evaluations of uniform random samples over the entire space $\mathcal{X}$ (to estimate the Lipschitz constant k), ECP evaluates a point only if it falls within a smaller subspace of $\mathcal{X}$ containing potential maximizers.

Notice that the acceptance region depends on the previously explored points until iteration time t and the current value of ε_t. As shown in the later analysis, the acceptance region exhibits intersting properties: it is non-decreasing with increasing values of ε_t at a given iteration t (for any time step t and any $u \leq v$, we have $\mathcal{A}_{\varepsilon_t=u,t} \subseteq \mathcal{A}_{\varepsilon_t=v,t}$, as shown in Lemma 1), and it is non-increasing with increasing time steps t for a given value of ε_t (for any $\varepsilon_t = \varepsilon$ and any $t_1 \leq t_2$, we have $\mathcal{A}_{\varepsilon_t=\varepsilon,t_2} \subseteq \mathcal{A}_{\varepsilon_t=\varepsilon,t_1}$, as shown in Proposition 3). Therefore, the smaller the ε_t and the larger the time t, the more points are rejected, as verified by the increasing upper bound on the rejection probability in Proposition 4. To balance this potential growth in rejections, ε_t is designed to continuously grow by a multiplicative constant $\tau_{n,d} > 1$, both to include more potential points and to mitigate the risk of exponential growth in rejections as the time step t increases, which ensures both a fast algorithm and guaranteed convergence, with polynomial computational complexity, as shown in Theorem 1.

Remark 1. (Extension to other smoothness) *The proposed framework can be generalized to encompass a broad class of globally and locally smooth functions by making slight modifications to the decision rule. As an example, consider the set of functions analyzed by Munos et al. (2014), characterized by a unique maximizer $x^\star$ and satisfying $f(x^\star) - f(x) \leq \ell(x^\star, x)$ for all $x \in X$, where $\ell : X \times X \to \mathbb{R}^+$ is a semi-metric defining local smoothness around the maxima. By adapting Proposition 2, the decision rule for selecting X_{t+1} can be reformulated as testing whether* $\max_{i=1,\ldots,t} f(X_i) \leq \min_{i=1,\ldots,t} f(X_i) + \ell(X_{t+1}, X_i)$.

5 THEORETICAL ANALYSIS

In the following, we motivate the considered acceptance region, analyze the rejection growth and computational complexity, and derive the regret of ECP.

5.1 Acceptance Region Analysis

In this section, we motivate the proposed acceptance region and the design of the algorithm with respect to the growing ε_t. The acceptance region is inspired by the previously studied active subset of consistent functions in active learning (Dasgupta, 2011; Hanneke, 2011; Malherbe and Vayatis, 2017). Hence, we start by the definition of consistent functions.

Definition 3. (Consistent functions) *The active subset of Lipschitz functions, with a Lipschitz constant k, consistent with the black-box function f over $t \geq 1$ evaluated samples $(x_1, f(x_1)), \cdots, (x_t, f(x_t))$ is:* $\mathcal{F}_{k,t} \triangleq \{g \in \mathrm{Lip}(k) : \forall i \in \{1, \cdots, t\},\ g(x_i) = f(x_i)\}$.

Using the above definition of a consistent function, we define the subset of points that can maximize at least some function g within that subset of consistent functions and possibly maximize the target f.

Definition 4. (Potential Maximizers) *For a Lipschitz function f with a Lipschitz constant $k \geq 0$, let $\mathcal{F}_{k,t}$ be the set of consistent functions with respect to f, as defined in Definition 3. For any iteration $t \geq 1$, the set of potential maximizers is defined as follows:*

$$\mathcal{P}_{k,t} \triangleq \left\{ x \in \mathcal{X} : \exists g \in \mathcal{F}_{k,t} \text{ where } x \in \arg\max_{x \in \mathcal{X}} g(x) \right\}.$$

We can then show the relationship between the potential maximizers and our proposed acceptance region. But first, we demonstrate an important characteristic of our acceptance region, which is being a no-decreasing region, function of the value of ε_t, as shown in Appendix C.2.

Lemma 1. (Expanding with Respect to ε_t) *Let $u, v > 0$ be two values of ε_t such that $u \leq v$. Then, the set of potentially accepted points at time t corresponding to $\varepsilon_t = u$ is a subset of the set of actions at time t corresponding to $\varepsilon_t = v$, i.e., $\mathcal{A}_{\varepsilon_t=u,t} \subseteq \mathcal{A}_{\varepsilon_t=v,t}$.*

Therefore, for a fixed iteration t, by design of our algorithm, which increases ε_t, the acceptance region is non-decreasing. Using the above result in Lemma 1 and Lemma 3 from Appendix C.1, we derive in Appendix C.3 the relationship between our considered acceptance region and the set of potential maximizers, summarized in the following proposition.

Proposition 2. (Potential Optimality) *For any iteration t, if $\mathcal{P}_{k,t}$ denotes the set of potential maximizers of $f \in \mathrm{Lip}(k)$, as in Definition 3, and $\mathcal{A}_{\varepsilon_t,t}$ denotes our acceptance region, defined in Definition 2, then:* $\forall \varepsilon_t \leq k, \quad \mathcal{A}_{\varepsilon_t,t} \subseteq \mathcal{P}_{k,t}$, *and* $\forall \varepsilon_t > k, \quad \mathcal{P}_{k,t} \subseteq \mathcal{A}_{\varepsilon_t,t}$.

Therefore, from the above proposition, we conclude that starting with an arbitrarily small ε_1, smaller than or equal to the unknown Lipschitz constant k, ECP evaluates the function only over sampled points that are potential maximizers of the unknown function f. Furthermore, when ε_t reaches or exceeds k, i.e., $\varepsilon_t \geq k$, which is unavoidable with a growing number of evaluations, the acceptance space does not exclude any potential maximizer, as all potential maximizers remain within the acceptance condition, which is crucial to guarantee the no-regret property of ECP.

Beyond the aforementioned inclusions, it can be seen that both our acceptance region $\mathcal{A}_{\varepsilon_t,t}$ and the true set of potential maximizers $\mathcal{P}_{k,t}$ are functions of the time step t. In fact, the set of consistent functions $\mathcal{F}_{k,t}$ is non-increasing as the number of evaluations increases. Consequently, the set of potential maximizers of at least one of these functions, $\mathcal{P}_{k,t}$, also becomes non-increasing with an increasing number of evaluations. We show in Appendix C.4 that, in addition to the inclusions in Proposition 2, our acceptance region follows the same trend with increasing iteration steps t. That is, our acceptance region is a non-increasing region with respect to t, as provided in Proposition 3.

Proposition 3. (Shrinking with respect to t) *For any $\varepsilon_t = \varepsilon$, for any $t_1, t_2 \geq 1$ such that $t_1 \leq t_2$, we have $\mathcal{P}_{k,t_2} \subseteq \mathcal{P}_{k,t_1}$ and $\mathcal{A}_{\varepsilon_t=\varepsilon,t_2} \subseteq \mathcal{A}_{\varepsilon_t=\varepsilon,t_1}$.*

Thus, given some fixed value of $\varepsilon_t = \varepsilon$, as t increases, accepting points becomes increasingly difficult.

Now, rethinking Proposition 2, we note that for $\varepsilon_t \leq k$, only potential maximizers are evaluated. A curious reader might ask: if it is guaranteed that $\mathcal{A}_{\varepsilon_t,t} \subseteq \mathcal{P}_{k,t}$ for $\varepsilon_t \leq k$—meaning all accepted points are potential maximizers—*why expand the acceptance region?*

Both Proposition 2 and Proposition 3 illustrate the relationship between our acceptance region and the set of potential maximizers, and together they motivate the growth of ε_t over the iterations for two key reasons. First, as indicated in Proposition 2, increasing ε_t ensures that we do not overlook any potential maximizers. Specifically, it is only when ε_t reaches or exceeds k that all potential maximizers fall within the acceptance region, i.e., $\mathcal{P}_{k,t} \subseteq \mathcal{A}_{\varepsilon_t,t}$. Second, as stated in Proposition 3, for a fixed ε_t, the acceptance region is a non-increasing set with respect to t, which can lead to a growing probability of rejection—potentially in an exponential manner. To counteract this exponential growth, our algorithm increases the value of ε_t, thereby preventing the rejection rate from becoming unsustainable. In the following section, we further analyze the rejection probability of sampled points from the region $\mathcal{X}$ and derive guarantees on the likelihood of accepting points, which ensures the probabilistic termination of the ECP algorithm in polynomial time.

5.2 Rejection Growth Analysis and Computational Complexity

When ε_t increases within the same iteration t, it is scaled by a multiplicative factor $\tau_{n,d} > 1$ whenever growth is detected, i.e., ε_t becomes $\varepsilon_t \tau_{n,d}^{v_t}$, where v_t represents the number of growth detection within iteration t. Consider $\Delta = \max_{x \in \mathcal{X}} f(x) - \min_{x \in \mathcal{X}} f(x)$, λ the standard Lebesgue measure that generalizes the notion of volume of any open set, and $\Gamma(x) = \int_0^\infty t^{x-1} e^{-t}\, \mathrm{d}t$. In what follows, we characterize this rejection growth by providing an upper bound on the probability of rejection in Proposition 4, proved in Appendix C.5, function of $\varepsilon_1 > 0$, $\tau_{n,d} > 1$, with $\varepsilon_t \geq \varepsilon_1 \cdot \tau_{n,d}^{(t-1)}$, and v_t which depends on $C \geq 1$.

Proposition 4. (ECP Rejection Probability) *For any Lipschitz function f, let $(x_i)_{1 \leq i \leq t}$ be the previously evaluated points of ECP until time t, and let v_t the number of increases of ε_t at iteration t (i.e., the number of times we validate growth condition in iteration t). For any $x \in \mathcal{X}$, let $R(x, t, v_t)$ be the event of rejecting x at time $t+1$ after v_t growths, we have:*

$$\mathbb{P}\left(R(x, t+1, v_t)\right) \leq \frac{t(\sqrt{\pi}\Delta)^d}{\varepsilon_1^d \tau_{nd}^{(t-1)d} \tau_{n,d}^{v_t d} \Gamma(d/2+1) \lambda(\mathcal{X})}.$$

For completeness, although the original works do not provide this information, we note that by applying the same bounding techniques used in Proposition 4 and leveraging the law of total probability, one can derive the rejection bounds for both AdaLIPO and AdaLIPO+. We present these results in Proposition 7 in Appendix C.6. Unlike our approach, which does not require space-filling to estimate the true Lipschitz constant, they allocate a portion of uniform random sampling for this purpose, with probability p. Consequently, their rejection upper bound is reduced by this factor, let $R(x, t)$ be the event of rejecting x at time $t+1$, then its probability is upperbounded as $\mathbb{P}\left(R(x, t+1)\right) \leq \frac{(1-p)t(\sqrt{\pi}\Delta)^d}{k_t^d \Gamma(d/2+1) \lambda(\mathcal{X})}$. In such case, it can

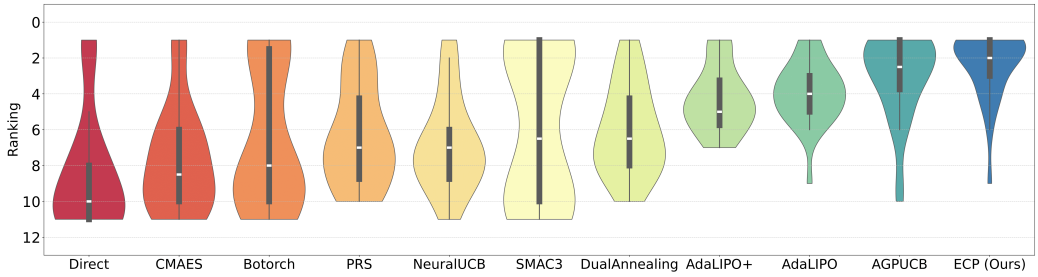


Figure 3: Violin plots showing the ranking distributions of diverse optimization algorithms, ordered by increasing median ranking, across 30 non-convex, multi-dimensional synthetic and real-world problems. The results are based on a budget of $n = 50$ for the different algorithms, with maxima averaged over 100 repetitions.

be seen that the upperbound of ECP is the closest to LIPO, which assumes the knowledge of k.

Remark 2. *If the exact value of k is known, then for any choice of $C \geq 1$, setting $\varepsilon_1 = k$ and $\tau_{n,d} = 1$ (in which case the choice of C becomes irrelevant since ε_t does not increase when multiplied by one), the ECP algorithm recovers the exact method used in LIPO (Malherbe and Vayatis, 2017), which assumes knowledge of k and is known to achieve optimal rates for known Lipschitz constants. Furthermore, our algorithm achieves the same rejection rate of LIPO, as shown in Theorem 1 in (Serré et al., 2024), for $\varepsilon_1 = k$ and $\tau_{n,d} = 1$.*

Increasing the values of ε_1 and $\tau_{n,d}$ causes the rejection probability to approach zero, reducing the algorithm to a pure random search and undermining the efficiency of function evaluations. Therefore, smaller values for both ε_1 and $\tau_{n,d}$ are required, which leads to a higher rejection rate and creates a trade-off between time complexity and output quality. We mitigate this by introducing a constant $C > 1$, allowing the use of smaller ε_1 and $\tau_{n,d}$ while preserving the algorithm's speed by adaptively increasing ε_t (when needed) in response to high rejection rates. ε_t grows deterministically, ensuring that $\varepsilon_t \geq \varepsilon_1 \tau_{n,d}^{t-1}$, and stochastically when growth conditions are met. By multiplying ε_t by $\tau_{n,d} > 1$ during rejection growth, even with small values of $\tau_{n,d}$ and ε_1, we guarantee the eventual acceptance of a point, proven in Appendix C.7.

Corollary 1. (Likely Acceptance) *There exists a maximum finite number of increases, v, independent of the iteration t, function of n and d, such that the probability of acceptance is at least $1/2$,*

$$v = \left\lceil \frac{1}{d} \log_{\tau_{n,d}} \left(\frac{2n(\sqrt{\pi}\Delta)^d}{\varepsilon_1^d \Gamma(d/2+1)\lambda(\mathcal{X})} \right) \right\rceil .$$

In the following we characterize this growth and show in Appendix C.8 that it maintains a linear growth in t, with coefficient $C > 1$, which ensures a fast algorithm.

Proposition 5. (Rejection Growth) *For any iteration step $t \geq 1$, the growth condition used in ECP, with a constant $C > 1$, ensures $h_{t+1} \leq (t+1) \cdot C$. Otherwise, ε_t grows and h_{t+1} is reset to zero.*

While algorithms like AdaLIPO may fall into an infinite loop of rejections, which happens with growing t, as the acceptance of a point gets harder, the result above is crucial in showing that the growth condition in ECP, i.e., $h_{t+1} - h_t > C$, prevents super-linear rejection growth by continuously increasing ε_t for the same iteration t. This increase makes point acceptance more likely in that iteration, as demonstrated in Lemma 1. The growth of ε_t continues until a point is accepted, thereby avoiding the infinite sampling loop.

By controlling the growth in the number of rejections, we can establish an upper bound on the worst-case computational complexity of ECP for a fixed evaluation budget of n. This result is formalized in the following theorem, shown in Appendix C.9.

Theorem 1. (ECP Computational Complexity) *Consider the ECP algorithm tuned with any $\varepsilon_1 > 0$, any $\tau_{n,d} > 1$, and any constant $C > 1$. Then, for any function $f \in \mathcal{F}$ and any budget $n \in \mathbb{N}^\star$, with a probability at least $1 - \frac{1}{2^C - 1}$, the computational complexity of ECP is at most $\mathcal{O}\left(\frac{n^2 C}{d} \log_{\tau_{n,d}} \left(\frac{n\Delta^d}{\varepsilon_1^d \Gamma(d/2+1)\lambda(\mathcal{X})} \right) \right)$.*

Remark 3. *Notice that a larger constant C implies less constraint on growth, as shown in Proposition 5, which leads to more patience before increasing ε_t. Furthermore, smaller values of $\varepsilon_1 > 0$ and $\tau_{n,d} > 1$ lead to a slower growth of ε_t, resulting in higher rejection rates, as shown in Proposition 4. Therefore, either increasing C, decreasing $\tau_{n,d}$, or decreasing ε_1 result in higher rejection rates and consequently potentailly more waiting time to accept a sampled point, thereby potentially increasing the computational complexity of the algorithm, as validated in Theorem 1.*

5.3 Regret Analysis

First, we define the $i^\star$ as the hitting time, after which ε_t reaches or overcomes the Lipschitz constant k.

Definition 5. (Hitting Time) *For the sequence* $(\varepsilon_i)_{i\in\mathbb{N}}$ *and the unknown Lipschitz constant* $k > 0$*, we can define* $i^\star \triangleq \min\{i \in \mathbb{N}^\star : \varepsilon_i \geq k\}$.

In the following lemma, we upper-bound the time t after which ε_t is guaranteed to reach or exceed k. This shows that for sufficiently large t, the event of reaching k is inevitable, with proof provided in Appendix C.10.

Lemma 2. (Hitting Time Upper-bound) *For any function* $f \in \mathrm{Lip}(k)$*, for any* $\tau_{n,d} > 1$ *and any* $\varepsilon_1 > 0$*, the hitting time* $i^\star$ *is upperbounded as follows:*

$$\forall \varepsilon_1 > 0, \quad i^\star \leq \max\left(\left\lceil \log_{\tau_{n,d}}\left(\frac{k}{\varepsilon_1}\right)\right\rceil, 1\right).$$

Unlike other optimization methods that do not guarantee no-regret, such as AdaLIPO+ (due to decaying exploration) or evolutionary algorithms like CMA-ES, similar to AdaLIPO, ECP is a no-regret algorithm over Lipschitz functions, as shown in Appendix C.11.

Theorem 2. (No-regret) *For any unkown Lipschitz constant* k*, the ECP algorithm tuned with any hyper-parameter* $\varepsilon_1 > 0$ *and using any geometric growth hyper-parameter* $\tau_{n,d} > 1$*, and* $C > 1$ *converges in probability to the exact maximum, i.e.*

$$\forall f \in \mathrm{Lip}(k),\ \mathcal{R}_{\mathrm{ECP},f}(n) \xrightarrow{p} 0.$$

If the function is Lipschitz, even with an unknown Lipschitz constant, similar to previous global optimization methods (Malherbe et al., 2016; Malherbe and Vayatis, 2017), we demonstrate in Appendix C.12 that ECP consistently outperforms or matches PRS.

Proposition 6. (ECP Faster than PRS) *Consider the ECP algorithm tuned with any initial value* $\varepsilon_1 > 0$*, any constant* $\tau_{n,d} > 1$*, and any constant* $C > 1$*. Then, for any* $f \in \mathrm{Lip}(k)$ *and* $n \geq i^\star$*,*

$$\mathbb{P}\left(\max_{i=i^\star,\cdots,n} f(x_i) \geq y\right) \geq \mathbb{P}\left(\max_{i=i^\star,\cdots,n} f(x'_i) \geq y\right)$$

for all $y \geq \max_{i=1,\cdots,i^\star-1} f(x_i)$*, where* $x_1, \cdots, x_n$ *are* n *evaluated points by ECP and* $x'_{i^\star}, \cdots, x'_n$ *are* n *independent uniformly distributed points over* $\mathcal{X}$*.*

Finally, the following theorem, shown in Appendix C.13, upperbounds the finite-time ECP regret.

Theorem 3. (Regret Upper Bound) *Consider ECP tuned with any* $\varepsilon_1 > 0$*, any* $\tau_{n,d} > 1$*, and any* $C > 1$*. Then, for any non-constant* $f \in \mathrm{Lip}(k)$*, with some unknown Lipschitz constant* k*, any* $n \in \mathbb{N}^\star$ *and* $\delta \in (0,1)$*, we have with probability at least* $1-\delta$*,*

$$\mathcal{R}_{ECP,f}(n) \leq \mathrm{diam}(\mathcal{X}) \cdot i^{\star\frac{1}{d}} \cdot k \cdot \left(\frac{\ln(\frac{1}{\delta})}{n}\right)^{\frac{1}{d}}.$$

With large values of $\tau_{n,d}$ and ε_1, $i^\star$ (bounded in Lemma 2) tends to one, and the ECP regret bound exactly recovers the finite-time upper bound of LIPO Malherbe and Vayatis (2017), which, unlike our method, requires knowledge of k, see Table 5. From the above theorem, it can be seen that ECP achieves the minimax optimal rate of $\mathcal{O}(kn^{-\frac{1}{d}})$, matching the lower bound $\Omega(kn^{-1/d})$ provided by Proposition 1.

6 NUMERICAL ANALYSIS

Since we do not assume knowledge of the Lipschitz constant, we consider benchmarks that do not require that. We compare our approach against 10 benchmarks, including AdaLIPO (Malherbe and Vayatis, 2017), AdaLIPO+ (Serré et al., 2024), PRS (Zabinsky, 2013), DIRECT (Jones et al., 1993), DualAnnealing (Xiang et al., 1997), CMA-ES (Hansen and Ostermeier, 1996; Hansen, 2006; Hansen et al., 2019), Botorch (Balandat et al., 2020), SMAC3 (Lindauer et al., 2022), A-GP-UCB (Berkenkamp et al., 2019), and NeuralUCB (Zhou et al., 2020). Details can be found in Appendix D.

We evaluate the benchmarks and our proposed method on various global optimization problems using both synthetic and real-world datasets. The synthetic functions were designed to challenge global optimization methods due to their non-convex curvatures (Molga and Smutnicki, 2005; Surjanovic and Bingham, 2013). Some of the 2D functions are shown in Fig. 1 for reference. For the real-world datasets, we follow the same set of global optimization problems considered in (Malherbe et al., 2016; Malherbe and Vayatis, 2017) drawn from (Frank, 2010), including Auto-MPG, Breast Cancer Wisconsin, Concrete Slump Test, Housing, and Yacht Hydrodynamics, optimizing the hyper-parameters of a Gaussian kernel ridge regression. The implementation of ECP and the considered objectives is publicly available at https://github.com/fouratifares/ECP.

We fix the same budget n for all the methods and report the maximum achieved value over the rounds, averaged over 100 repetitions, with reported standard deviations. Across all optimization problems, we fix $\varepsilon_1 = 10^{-2}$ and use $\tau_{n,d} = \max\left(1 + \frac{1}{nd}, \tau\right)$, with $\tau = 1.001$ and $C = 10^3$ (discussion and ablation of these parameters can be found in Appendix E). Results for $n = 50$ are presented in Table 1.

ECP demonstrates outstanding performance across benchmarks, achieving the highest values (Top-1) in 13 problems, while other approaches achieved at most 10 out of the 30 problems. We further illustrate the ranking distribution of the 11 algorithms in the violin plot in Fig. 3 across all the objective functions.

Problems	PRS	DIRECT	CMA-ES	DualAnnealing	NeuralUCB	AdaLIPO	AdaLIPO+	Botorch	SMAC3	A-GP-UCB	ECP (Ours)
autoMPG 2D	-30.31 (5.69)	-27.85 (0.00)	-25.77 (8.14)	-28.36 (3.81)	-30.75 (6.44)	-25.58 (1.67)	-26.34 (2.02)	**-23.10 (0.00)**	-23.32 (0.00)	**-23.10 (0.00)**	-25.17 (1.50)
breastCancer 2D	-0.08 (0.01)	-0.08 (0.00)	**-0.07 (0.01)**	-0.08 (0.01)	-0.08 (0.00)	**-0.07 (0.00)**	**-0.07 (0.00)**	**-0.07 (0.00)**	**-0.07 (0.00)**	**-0.07 (0.00)**	**-0.07 (0.00)**
concrete 2D	-27.85 (2.84)	-28.31 (0.00)	-26.05 (6.30)	-27.07 (1.72)	-27.75 (2.84)	-25.67 (0.91)	-26.07 (1.13)	-24.50 (1.25)	**-24.42 (0.00)**	-24.45 (0.14)	-25.33 (0.62)
housing 2D	-13.60 (0.62)	-13.79 (0.00)	-13.14 (0.91)	-13.40 (0.49)	-13.68 (0.68)	-12.99 (0.16)	-13.09 (0.22)	**-12.73 (0.00)**	-12.75 (0.00)	**-12.73 (0.00)**	-12.98 (0.13)
yacht 2D	-67.54 (6.85)	-64.46 (0.00)	-61.48 (7.21)	-65.64 (5.69)	-67.75 (7.73)	-59.78 (1.04)	-60.75 (1.58)	**-58.07 (0.00)**	-58.40 (0.00)	**-58.07 (0.00)**	-60.08 (1.50)
Ackley 2D	-4.92 (1.48)	-4.92 (0.00)	-8.69 (5.47)	-4.72 (1.72)	-5.26 (1.62)	-2.08 (1.19)	-2.37 (1.42)	-6.39 (2.11)	-4.89 (0.00)	-1.39 (2.35)	**-1.38 (0.80)**
Bukin 2D	-21.09 (10.09)	-47.28 (0.00)	-12.51 (13.60)	-19.43 (10.22)	-18.92 (9.11)	-14.54 (6.95)	-15.22 (6.97)	-32.41 (16.33)	**-0.91 (0.00)**	-8.40 (8.81)	-11.33 (5.50)
Camel 2D	0.89 (0.13)	0.99 (0.00)	0.90 (0.26)	0.90 (0.14)	0.97 (0.06)	1.01 (0.02)	0.99 (0.05)	0.72 (0.32)	**1.02 (0.00)**	1.01 (0.10)	**1.02 (0.01)**
Cross-in-tray 2D	1.99 (0.07)	1.96 (0.00)	1.76 (0.18)	2.00 (0.07)	1.95 (0.10)	2.01 (0.07)	2.02 (0.07)	1.94 (0.10)	1.88 (0.00)	2.00 (0.09)	**2.03 (0.06)**
Damavandi 2D	-3.57 (1.56)	-9.26 (0.00)	-4.37 (5.71)	-3.18 (1.26)	-4.17 (2.17)	-2.55 (0.57)	-2.59 (0.56)	-5.53 (3.27)	**-2.02 (0.00)**	-2.83 (3.42)	-2.24 (0.29)
Drop-wave 2D	0.73 (0.13)	0.14 (0.00)	0.58 (0.24)	0.75 (0.13)	0.70 (0.20)	0.74 (0.12)	**0.76 (0.12)**	0.66 (0.16)	0.20 (0.00)	0.70 (0.19)	**0.76 (0.12)**
Easom 2D	0.06 (0.18)	0.00 (0.00)	0.04 (0.18)	0.05 (0.12)	0.07 (0.21)	0.05 (0.15)	0.06 (0.16)	0.02 (0.09)	0.00 (0.00)	**0.13 (0.33)**	0.06 (0.15)
Egg-holder 2D	61.11 (11.57)	54.74 (0.00)	21.38 (22.00)	64.77 (11.85)	64.26 (11.90)	59.32 (10.44)	62.99 (12.71)	60.24 (10.02)	**84.71 (0.00)**	62.49 (13.57)	69.91 (11.70)
Griewank 2D	-0.26 (0.13)	-1.06 (0.00)	-0.43 (0.28)	-0.27 (0.14)	-0.26 (0.13)	-0.27 (0.12)	-0.27 (0.12)	-0.40 (0.21)	-0.43 (0.00)	**-0.13 (0.11)**	-0.25 (0.13)
Himmelblau 2D	-2.96 (3.12)	-3.94 (0.00)	-2.32 (5.95)	-2.96 (2.76)	-3.17 (2.85)	-1.25 (1.40)	-1.56 (1.58)	-7.10 (7.16)	**-0.08 (0.00)**	-0.96 (4.03)	-0.74 (0.82)
Holder 2D	14.44 (3.42)	8.83 (0.00)	6.61 (4.98)	14.69 (3.45)	14.33 (3.48)	15.53 (3.19)	15.22 (3.05)	15.97 (1.34)	0.78 (0.00)	16.08 (4.45)	**17.03 (2.17)**
Langermann 2D	2.92 (0.76)	**3.98 (0.00)**	1.61 (1.13)	2.73 (0.70)	2.63 (1.00)	2.71 (0.82)	2.69 (0.83)	2.30 (0.94)	2.63 (0.00)	2.80 (1.01)	2.32 (1.10)
Levy 2D	-3.87 (3.56)	-6.56 (0.00)	-35.73 (45.40)	-2.60 (2.33)	-4.48 (4.18)	-1.63 (1.04)	-2.27 (1.83)	-7.5 (7.34)	-4.18 (0.00)	-1.67 (4.82)	**-0.80 (0.49)**
Michalewicz 2D	1.11 (0.28)	1.36 (0.00)	1.20 (0.37)	1.08 (0.27)	1.06 (0.27)	1.28 (0.30)	1.21 (0.28)	0.98 (0.28)	1.00 (0.00)	1.35 (0.41)	**1.38 (0.29)**
Rastrigin 2D	-6.86 (3.52)	-9.63 (0.00)	-12.69 (9.86)	-5.98 (3.58)	-6.73 (3.62)	-6.75 (3.39)	-6.66 (3.45)	-9.9 (5.65)	-38.22 (0.00)	**-5.52 (3.51)**	**-5.52 (2.93)**
Schaffer 2D	**-0.01 (0.01)**	**-0.01 (0.00)**	**-0.01 (0.01)**	**-0.01 (0.01)**	**-0.01 (0.01)**	**-0.01 (0.01)**	**-0.01 (0.01)**	**-0.01 (0.02)**	-0.94 (0.00)	**-0.01 (0.01)**	**-0.01 (0.01)**
Schubert 2D	8.28 (4.51)	1.18 (0.00)	4.46 (4.10)	8.36 (4.87)	7.60 (4.15)	7.92 (4.44)	7.90 (4.82)	5.26 (3.55)	3.79 (0.00)	**9.80 (6.40)**	7.80 (4.46)
Colville 3D	-0.26 (0.26)	**-0.06 (0.00)**	-1.63 (3.22)	-0.33 (0.42)	-0.33 (0.32)	-0.20 (0.16)	-0.26 (0.32)	-0.76 (1.02)	-0.69 (0.00)	-0.15 (0.16)	-0.17 (0.14)
Hartmann 3D	3.42 (0.31)	3.51 (0.00)	3.57 (0.36)	3.35 (0.32)	3.64 (0.17)	3.73 (0.11)	3.68 (0.17)	**3.86 (0.00)**	2.89 (0.00)	3.73 (0.27)	3.79 (0.04)
Hartmann 6D	1.77 (0.56)	0.42 (0.00)	1.99 (0.58)	1.69 (0.53)	1.77 (0.52)	1.94 (0.52)	1.93 (0.49)	**3.21 (0.26)**	2.54 (0.00)	2.75 (0.60)	2.01 (0.43)
Rosenbrock 3D	-0.48 (0.26)	-0.54 (0.00)	-0.40 (0.34)	-0.42 (0.26)	-0.32 (0.16)	-0.36 (0.18)	-0.37 (0.18)	-0.62 (0.32)	**-0.10 (0.00)**	-0.24 (0.25)	-0.16 (0.08)
Perm 10D	-0.13 (0.14)	nan (nan)	-1.28 (3.10)	-0.15 (0.14)	-0.13 (0.12)	-0.11 (0.12)	-0.11 (0.10)	-0.26 (0.23)	**-0.01 (0.00)**	-0.13 (0.19)	-0.08 (0.07)
Perm 20D	-2.52 (2.21)	nan (nan)	-44.41 (124.04)	-2.99 (3.11)	-3.02 (4.65)	-2.24 (2.24)	-2.46 (2.62)	-7.21 (7.06)	-3.81 (0.00)	-2.91 (4.25)	**-1.59 (1.53)**
Powell 100D	3.20 (0.29)	nan (nan)	2.76 (0.62)	3.24 (0.30)	3.27 (0.33)	3.36 (0.34)	3.33 (0.33)	3.30 (0.00)	3.42 (0.00)	3.18 (0.29)	**3.64 (0.34)**
Powell 1000D	**0.23 (0.01)**	nan (nan)	**0.23 (0.01)**	**0.23 (0.01)**	**0.23 (0.01)**	**0.23 (0.01)**	**0.23 (0.01)**	**0.23 (0.01)**	0.22 (0.00)	**0.23 (0.01)**	**0.23 (0.01)**
# Top-1	2	3	3	2	2	3	4	8	9	10	**13**

Table 1: Comparison of the maximum returned value using a budget of $n = 50$ for the different algorithms, averaged over 100 repetitions with standard deviations reported. The ECP hyperparameters are fixed across all problems: $\varepsilon_1 = 10^{-2}$, $\tau_{n,d} = \max\left(1 + \frac{1}{nd}, \tau\right)$, with $\tau = 1.001$, and $C = 10^3$. nan indicates an error related to insufficient budget.

Clearly, ECP shows the highest median ranking across the objectives and the highest tendency towards top rankings. Notably, it recovers or outperforms all other algorithms on the highest-dimensional problems (Perm 20D, Powell 100D, and Powell 1000D). Additionally, as shown in Table 2 for half the budget and Table 3 for double the budget, ECP outperforms all the considered Lipschitz, bandit, and evolutionary approaches.

While the main focus of this work is on expensive functions and small budgets, it is evident from Table 3 and Table 4 in Appendix A that ECP continues to perform well even for larger budgets. However, while larger budgets do not disadvantage ECP, its performance advantage diminishes as other global optimization methods, such as CMA-ES, are sometimes able to find better approximations in these larger-budget settings. Additionally, Table 4 highlights a key difference between ECP and similar methods for Lipschitz functions, such as AdaLIPO and AdaLIPO+. These methods can encounter infinite loops as their complexity increases with larger budgets. In contrast, ECP avoids such problems by adaptively increasing the acceptance region through the gradual expansion of ε_t, which validates our earlier observations following Proposition 5.

7 DISCUSSION

Several global optimization approaches rely on surrogate models. While surrogate-based methods excel at modeling complex function landscapes, they introduce significant computational overhead and risk overfitting in low-budget scenarios. With a limited budget, learning an accurate surrogate model becomes challenging, leading to poor sampling and suboptimal performance. In contrast, ECP is specifically designed for low-budget settings, which are common when dealing with expensive functions. Although large budgets do not degrade ECP's performance, when computational resources permit a high number of function evaluations, methods that allocate some of these evaluations to learning the function's smoothness or constructing a surrogate model may perform better.

ECP provides theoretical guarantees under the sole assumption of global Lipschitz continuity, without requiring prior knowledge of the Lipschitz constant. While this assumption is minimal compared to others, it may not hold in all real-world scenarios. Nevertheless, empirically, ECP outperforms several established algorithms across a range of non-convex optimization problems, including cases where the Lipschitz continuity assumption is not strictly satisfied.

8 CONCLUSION

We introduce ECP, a global optimization algorithm for black-box functions with unknown Lipschitz constants. Our theoretical analysis shows that ECP is no-regret as evaluations increase and meets minimax optimal regret bounds within a finite budget. Empirical results across diverse non-convex, multi-dimensional optimization tasks demonstrate that ECP outperforms state-of-the-art methods.

References

Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(Nov):397–422, 2002.

Maximilian Balandat, Brian Karrer, Daniel Jiang, Samuel Daulton, Ben Letham, Andrew G Wilson, and Eytan Bakshy. Botorch: A framework for efficient monte-carlo bayesian optimization. *Advances in neural information processing systems*, 33:21524–21538, 2020.

Peter L. Bartlett, Victor Gabillon, and Michal Valko. A simple parameter-free and adaptive approach to optimization under a minimal local smoothness assumption. In *Proceedings of the 30th International Conference on Algorithmic Learning Theory*, volume 98 of *Proceedings of Machine Learning Research*, pages 184–206. PMLR, 22–24 Mar 2019. URL `https://proceedings.mlr.press/v98/bartlett19a.html`.

Felix Berkenkamp, Angela P. Schoellig, and Andreas Krause. No-regret bayesian optimization with unknown hyperparameters. *Journal of Machine Learning Research*, 20(50):1–24, 2019. URL `http://jmlr.org/papers/v20/18-213.html`.

Samuel H Brooks. A discussion of random methods for seeking maxima. *Operations research*, 6(2):244–251, 1958.

Sébastien Bubeck, Rémi Munos, Gilles Stoltz, and Csaba Szepesvári. X-armed bandits. *Journal of Machine Learning Research*, 12(5), 2011.

Adam D Bull. Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research*, 12(10), 2011.

Lichang Chen, Jiuhai Chen, Tom Goldstein, Heng Huang, and Tianyi Zhou. InstructZero: Efficient instruction optimization for black-box large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 6503–6518. PMLR, 21–27 Jul 2024. URL `https://proceedings.mlr.press/v235/chen24e.html`.

Sanjoy Dasgupta. Two faces of active learning. *Theoretical computer science*, 412(19):1767–1781, 2011.

Sarah Filippi, Olivier Cappe, Aurélien Garivier, and Csaba Szepesvári. Parametric bandits: The generalized linear case. *Advances in neural information processing systems*, 23, 2010.

D Finkel and Carl Tim Kelley. Convergence analysis of the direct algorithm. Technical report, North Carolina State University. Center for Research in Scientific Computation, 2004.

Christodoulos A Floudas and Panos M Pardalos. Recent advances in global optimization. 2014.

Fares Fourati, Vaneet Aggarwal, Christopher Quinn, and Mohamed-Slim Alouini. Randomized greedy learning for non-monotone stochastic submodular maximization under full-bandit feedback. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pages 7455–7471. PMLR, 25–27 Apr 2023. URL `https://proceedings.mlr.press/v206/fourati23a.html`.

Fares Fourati, Vaneet Aggarwal, and Mohamed-Slim Alouini. Stochastic q-learning for large discrete action spaces. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 13734–13759. PMLR, 21–27 Jul 2024a. URL `https://proceedings.mlr.press/v235/fourati24a.html`.

Fares Fourati, Mohamed-Slim Alouini, and Vaneet Aggarwal. Federated combinatorial multi-agent multi-armed bandits. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 13760–13782. PMLR, 21–27 Jul 2024b. URL `https://proceedings.mlr.press/v235/fourati24b.html`.

Fares Fourati, Christopher John Quinn, Mohamed-Slim Alouini, and Vaneet Aggarwal. Combinatorial stochastic-greedy bandit. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 12052–12060, 2024c.

Andrew Frank. Uci machine learning repository. *http://archive. ics. uci. edu/ml*, 2010.

Peter I Frazier. A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1861–1870. PMLR, 10–15 Jul 2018. URL `https://proceedings.mlr.press/v80/haarnoja18b.html`.

Steve Hanneke. Rates of convergence in active learning. *The Annals of Statistics*, pages 333–361, 2011.

Nikolaus Hansen. The cma evolution strategy: a comparing review. *Towards a new evolutionary computation: Advances in the estimation of distribution algorithms*, pages 75–102, 2006.

Nikolaus Hansen and Andreas Ostermeier. Adapting arbitrary normal mutation distributions in evo-

lution strategies: The covariance matrix adaptation. In *Proceedings of IEEE international conference on evolutionary computation*, pages 312–317. IEEE, 1996.

Nikolaus Hansen, Youhei Akimoto, and Petr Baudis. CMA-ES/pycma on Github. Zenodo, DOI:10.5281/zenodo.2559634, February 2019. URL https://doi.org/10.5281/zenodo.2559634.

Waltraud Huyer and Arnold Neumaier. Global optimization by multilevel coordinate search. *Journal of Global Optimization*, 14:331–355, 1999.

Donald R Jones and Joaquim RRA Martins. The direct algorithm: 25 years later. *Journal of global optimization*, 79(3):521–566, 2021.

Donald R Jones, Cary D Perttunen, and Bruce E Stuckman. Lipschitzian optimization without the lipschitz constant. *Journal of optimization Theory and Applications*, 79:157–181, 1993.

Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13:455–492, 1998.

Kenji Kawaguchi, Yu Maruyama, and Xiaoyu Zheng. Global continuous optimization with error bound and fast convergence. *Journal of Artificial Intelligence Research*, 56:153–195, 2016.

Salma Kharrat, Fares Fourati, and Marco Canini. Acing: Actor-critic for instruction learning in black-box large language models. *arXiv preprint arXiv:2411.12736*, 2024.

Scott Kirkpatrick, C Daniel Gelatt Jr, and Mario P Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

Robert Kleinberg, Aleksandrs Slivkins, and Eli Upfal. Multi-armed bandits in metric spaces. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 681–690, 2008.

John Langford and Tong Zhang. The epoch-greedy algorithm for contextual multi-armed bandits. *Advances in neural information processing systems*, 20 (1):96–1, 2007.

Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

Xiaoqiang Lin, Zhaoxuan Wu, Zhongxiang Dai, Wenyang Hu, Yao Shu, See-Kiong Ng, Patrick Jaillet, and Bryan Kian Hsiang Low. Use your INSTINCT: INSTruction optimization for LLMs usIng neural bandits coupled with transformers. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 30317–30345. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/lin24r.html.

Marius Lindauer, Katharina Eggensperger, Matthias Feurer, André Biedenkapp, Difan Deng, Carolin Benjamins, Tim Ruhkopf, René Sass, and Frank Hutter. Smac3: A versatile bayesian optimization package for hyperparameter optimization. *Journal of Machine Learning Research*, 23(54): 1–9, 2022. URL http://jmlr.org/papers/v23/21-0888.html.

Cédric Malherbe and Nicolas Vayatis. Global optimization of Lipschitz functions. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2314–2323. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/malherbe17a.html.

Cedric Malherbe, Emile Contal, and Nicolas Vayatis. A ranking approach to global optimization. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1539–1547, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/malherbe16.html.

Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

Marcin Molga and Czesław Smutnicki. Test functions for optimization needs. *Test functions for optimization needs*, 101:48, 2005.

Rémi Munos. Optimistic optimization of a deterministic function without the knowledge of its smoothness. *Advances in neural information processing systems*, 24, 2011.

Rémi Munos et al. From bandits to monte-carlo tree search: The optimistic principle applied to optimization and planning. *Foundations and Trends® in Machine Learning*, 7(1):1–129, 2014.

Fernando Nogueira. Bayesian Optimization: Open source constrained global optimization tool for Python, 2014–. URL https://github.com/bayesian-optimization/BayesianOptimization.

Masahiro Nomura and Masashi Shibata. Cmaes: A simple yet practical python library for cma-es. *arXiv preprint arXiv:2402.01373*, 2024.

OpenAI. Chatgpt, 2023. URL https://chat.openai.com.

Panos M Pardalos. *Handbook of global optimization*, volume 2. Springer Science & Business Media, 2013.

SA Piyavskii. An algorithm for finding the absolute extremum of a function. *USSR Computational Mathematics and Mathematical Physics*, 12(4):57–67, 1972.

Philippe Preux, Rémi Munos, and Michal Valko. Bandits attack function optimization. In *2014 IEEE congress on evolutionary computation (CEC)*, pages 2245–2252. IEEE, 2014.

Gaëtan Serré, Perceval Beja-Battais, Sophia Chirrane, Argyris Kalogeratos, and Nicolas Vayatis. Lipo+: Frugal global optimization for lipschitz functions. *arXiv preprint arXiv:2406.19723*, 2024.

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando de Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016. doi: 10.1109/JPROC.2015.2494218.

Bruno O Shubert. A sequential method seeking the global maximum of a function. *SIAM Journal on Numerical Analysis*, 9(3):379–388, 1972.

Jörg Stork, Agoston E Eiben, and Thomas Bartz-Beielstein. A new taxonomy of global optimization algorithms. *Natural Computing*, 21(2):219–242, 2022.

S. Surjanovic and D. Bingham. Virtual library of simulation experiments: Test functions and datasets. Retrieved October 10, 2024, from `http://www.sfu.ca/~ssurjano`, 2013.

Richard S Sutton. Reinforcement learning: An introduction. *A Bradford Book*, 2018.

Aimo Törn and Antanas Žilinskas. *Global optimization*, volume 350. Springer, 1989.

Constantino Tsallis. Possible generalization of boltzmann-gibbs statistics. *Journal of statistical physics*, 52:479–487, 1988.

Constantino Tsallis and Daniel A Stariolo. Generalized simulated annealing. *Physica A: Statistical Mechanics and its Applications*, 233(1-2):395–406, 1996.

Michal Valko, Nathaniel Korda, Rémi Munos, Ilias Flaounas, and Nelo Cristianini. Finite-time analysis of kernelised contextual bandits. *arXiv preprint arXiv:1309.6869*, 2013.

Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 17(3):261–272, 2020.

Yang Xiang, DY Sun, W Fan, and XG Gong. Generalized simulated annealing algorithm and its application to the thomson model. *Physics Letters A*, 233 (3):216–220, 1997.

Zelda B Zabinsky. *Stochastic adaptive search for global optimization*, volume 72. Springer Science & Business Media, 2013.

Dongruo Zhou, Lihong Li, and Quanquan Gu. Neural contextual bandits with UCB-based exploration. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11492–11502. PMLR, 13–18 Jul 2020. URL `https://proceedings.mlr.press/v119/zhou20a.html`.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes, Section 3 and Section 4]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes, Section 5]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes, supplemental material (https://github.com/fouratifares/ECP)]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes, Section 3]
   (b) Complete proofs of all theoretical results. [Yes, Appendix C]
   (c) Clear explanations of any assumptions. [Yes, Section 3]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes, Appendix D and supplemental material (https://github.com/fouratifares/ECP)]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes, Appendix D and Appendix E]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes, Appendix D]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes, Appendix D]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes, Appendix D]
   (b) The license information of the assets, if applicable. [Yes, Appendix D]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Yes, supplemental material]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

A EXTENDED EMPIRICAL RESULTS FOR VARIOUS BUDGETS

We fix the same budget n for all the methods and report the maximum achieved value over the rounds, averaged over 100 repetitions, with reported standard deviations. Across all optimization problems, we fix $\varepsilon_1 = 10^{-2}$ and use $\tau_{n,d} = \max\left(1 + \frac{1}{nd}, \tau\right)$, with $\tau = 1.001$ and $C = 10^3$ (discussion and ablation of these parameters can be found in Appendix E). Results for $n = 50$ are presented in the main paper in Table 1. Results for $n = 25$ are presented in Table 2. Results for $n = 100$ are presented in Table 3. Results for $n = 300$ are presented in Table 4.

Objectives	PRS	DIRECT	CMA-ES	DualAnnealing	NeuralUCB	AdaLIPO	AdaLIPO+	ECP (Ours)
autoMPG	-33.17 (7.34)	-45.86 (0.00)	-31.41 (16.40)	-33.87 (7.47)	-33.15 (6.51)	-27.20 (2.74)	-29.24 (4.31)	**-26.69 (2.58)**
breastCancer	-0.09 (0.01)	-0.12 (0.00)	-0.08 (0.01)	-0.09 (0.02)	-0.09 (0.03)	**-0.07 (0.00)**	-0.08 (0.00)	**-0.07 (0.00)**
concrete	-29.75 (3.74)	-41.50 (0.00)	-28.73 (15.73)	-30.03 (4.60)	-29.48 (3.43)	-26.61 (1.83)	-27.42 (1.99)	**-26.33 (1.45)**
housing	-14.03 (0.82)	-16.81 (0.00)	-13.53 (1.73)	-14.17 (1.27)	-14.00 (0.77)	-13.23 (0.34)	-13.36 (0.44)	**-13.20 (0.31)**
yacht	-70.97 (8.81)	-84.59 (0.00)	-67.18 (10.02)	-72.33 (9.86)	-71.22 (8.12)	**-62.24 (3.06)**	-63.93 (4.24)	-62.35 (2.75)
ackley	-6.25 (1.93)	-4.92 (0.00)	-10.97 (4.33)	-6.29 (2.36)	-6.27 (2.03)	-3.26 (1.40)	-3.82 (1.86)	**-2.69 (1.15)**
bukin	-28.82 (13.32)	-83.45 (0.00)	-24.93 (23.96)	-31.37 (14.29)	-31.50 (15.25)	-24.07 (11.03)	-25.59 (11.56)	**-23.01 (10.19)**
camel	0.73 (0.27)	-2.25 (0.00)	0.71 (0.34)	0.73 (0.27)	0.88 (0.17)	0.95 (0.10)	0.92 (0.14)	**0.99 (0.07)**
crossintray	1.94 (0.09)	1.96 (0.00)	1.74 (0.15)	1.93 (0.09)	1.91 (0.10)	**1.97 (0.09)**	1.94 (0.09)	**1.97 (0.10)**
damavandi	-5.35 (3.00)	-57.26 (0.00)	-11.83 (13.30)	-5.36 (3.63)	-5.72 (3.83)	-3.21 (1.39)	-3.96 (1.99)	**-2.57 (0.54)**
dropwave	0.65 (0.15)	0.14 (0.00)	0.48 (0.24)	0.62 (0.16)	0.54 (0.20)	0.64 (0.17)	**0.67 (0.16)**	**0.67 (0.16)**
easom	**0.05 (0.16)**	0.00 (0.00)	0.00 (0.00)	0.02 (0.06)	0.03 (0.13)	0.01 (0.08)	0.02 (0.09)	0.02 (0.09)
eggholder	55.37 (12.86)	1.17 (0.00)	19.78 (25.68)	55.43 (12.85)	55.04 (14.20)	55.28 (14.46)	53.89 (11.79)	**58.52 (13.37)**
griewank	-0.37 (0.16)	-1.56 (0.00)	-0.53 (0.31)	-0.39 (0.19)	-0.38 (0.18)	-0.36 (0.16)	-0.37 (0.16)	**-0.35 (0.19)**
himmelblau	-5.20 (5.39)	-5.83 (0.00)	-9.34 (17.93)	-6.84 (6.91)	-5.76 (4.83)	-2.83 (2.86)	-3.82 (4.04)	**-2.73 (2.30)**
holder	12.20 (3.85)	2.57 (0.00)	5.97 (4.56)	11.99 (3.98)	12.00 (4.19)	13.26 (3.73)	12.63 (4.27)	**15.18 (3.10)**
langermann	**2.32 (0.97)**	0.04 (0.00)	1.04 (1.01)	2.22 (0.87)	2.15 (0.97)	2.10 (0.98)	2.26 (0.94)	1.71 (1.08)
levy	-7.29 (6.85)	-22.20 (0.00)	-52.70 (45.67)	-6.18 (5.91)	-8.38 (7.82)	-3.85 (4.31)	-5.56 (5.78)	**-1.78 (1.44)**
michalewicz	0.97 (0.26)	**1.36 (0.00)**	0.97 (0.40)	0.92 (0.27)	0.89 (0.25)	1.07 (0.30)	0.97 (0.33)	1.01 (0.33)
rastrigin	-9.79 (5.41)	-60.41 (0.00)	-16.16 (8.96)	-9.73 (5.19)	-10.63 (5.08)	-9.28 (4.72)	-10.06 (4.89)	**-7.02 (3.97)**
schaffer	-0.02 (0.03)	**-0.01 (0.00)**	**-0.01 (0.01)**	**-0.01 (0.01)**	**-0.01 (0.01)**	**-0.01 (0.01)**	**-0.01 (0.01)**	**-0.01 (0.01)**
schubert	6.26 (4.50)	0.55 (0.00)	4.02 (4.13)	5.55 (4.28)	5.48 (4.12)	5.65 (3.92)	5.42 (4.14)	**7.27 (4.64)**
colville 4D	**-0.52 (0.51)**	nan (nan)	-9.65 (12.30)	-0.65 (0.66)	-0.64 (0.81)	-0.59 (0.56)	-0.70 (0.82)	-0.68 (0.80)
hartmann 3D	3.08 (0.47)	0.99 (0.00)	3.27 (0.60)	3.10 (0.45)	3.43 (0.35)	3.49 (0.30)	3.44 (0.33)	**3.63 (0.21)**
hartmann 6D	1.48 (0.61)	nan (nan)	**1.68 (0.56)**	1.29 (0.59)	1.44 (0.63)	1.61 (0.54)	1.52 (0.56)	1.51 (0.49)
rosenbrock 3D	-0.64 (0.32)	nan (nan)	-0.83 (0.62)	-0.63 (0.36)	-0.47 (0.29)	-0.50 (0.27)	-0.61 (0.29)	**-0.29 (0.23)**
Perm 10D	-0.23 (0.22)	nan (nan)	-1.25 (2.90)	-0.29 (0.25)	-0.22 (0.23)	-0.18 (0.16)	-0.24 (0.22)	**-0.15 (0.16)**
perm 20D	-5.38 (5.90)	nan (nan)	-73.29 (153.01)	-5.52 (7.49)	-4.90 (6.43)	-5.52 (7.97)	-4.83 (7.06)	**-3.76 (4.14)**
# Top-1	3	2	1	1	2	4	2	22

Table 2: Comparison of the maximum returned value using a budget of $n = 25$ for the different algorithms, averaged over 100 repetitions with standard deviations reported. The ECP hyperparameters are fixed across all problems: $\varepsilon_1 = 10^{-2}$, $\tau_{n,d} = \max\left(1 + \frac{1}{nd}, \tau\right)$, with $\tau = 1.001$, and $C = 10^3$. For AdaLIPO, we set $p = 0.1$, as specified in their paper. nan indicates an error related insufficient budget to return a value.

B EXTENDED RELATED WORKS

Several methods have been proposed for global optimization (Törn and Žilinskas, 1989; Pardalos, 2013; Floudas and Pardalos, 2014; Zabinsky, 2013; Stork et al., 2022). The most straightforward ones are non-adaptive exhaustive searches, such as brute-force methods, also known as grid search, which involves dividing the space into representative points and evaluating each one (Zabinsky, 2013). A stochastic version of grid search is Pure Random Search (PRS) (Brooks, 1958; Zabinsky, 2013), which uses random uniform sampling. While both of these approaches may work in certain situations, their non-adaptive nature makes them generally inefficient, particularly in low-budget settings, as they can lead to unnecessary function evaluations by failing to leverage previously discovered information (Zabinsky, 2013), as well as, failing to leverage any potential structure of the function.

To improve upon exhaustive search, several methods have been proposed to leverage previously discovered information, as well as potential structure of the objective function, such as Lipschitz continuity or smoothness (Shubert, 1972; Kleinberg et al., 2008; Bubeck et al., 2011; Munos, 2011; Preux et al., 2014; Kawaguchi et al., 2016; Bartlett et al., 2019). Some of these algorithms need the knowledge of the local smoothness such as HOO

Objectives	PRS	DIRECT	CMA-ES	DualAnnealing	NeuralUCB	AdaLIPO	AdaLIPO+	ECP (Ours)
autoMPG	-27.21 (2.86)	-24.46 (0.00)	**-23.15 (0.19)**	-25.76 (1.88)	-27.67 (3.01)	-24.42 (0.73)	-24.61 (0.89)	-24.11 (0.62)
breastCancer	-0.08 (0.00)	**-0.07 (0.00)**	**-0.07 (0.00)**	-0.08 (0.00)	**-0.07 (0.00)**	**-0.07 (0.00)**	**-0.07 (0.00)**	**-0.07 (0.00)**
concrete	-26.24 (1.21)	-25.56 (0.00)	**-24.46 (0.26)**	-25.90 (1.05)	-26.79 (1.57)	-25.22 (0.59)	-25.37 (0.59)	-24.92 (0.37)
housing	-13.22 (0.29)	-13.05 (0.00)	**-12.76 (0.08)**	-13.08 (0.25)	-13.35 (0.36)	-12.85 (0.07)	-12.89 (0.08)	-12.84 (0.07)
yacht	-63.71 (3.77)	-59.95 (0.00)	**-58.11 (0.12)**	-61.81 (2.48)	-64.87 (4.30)	-58.63 (0.39)	-59.17 (0.65)	-58.67 (0.39)
ackley	-4.23 (1.20)	-4.90 (0.00)	-6.21 (6.43)	-3.27 (1.17)	-4.09 (1.12)	-1.05 (0.71)	-1.17 (0.90)	**-0.71 (0.43)**
bukin	-16.09 (7.33)	-27.38 (0.00)	**-4.79 (3.89)**	-13.55 (7.94)	-15.41 (7.58)	-8.77 (4.21)	-9.68 (4.94)	-8.74 (3.99)
camel	0.96 (0.08)	0.99 (0.00)	0.84 (0.51)	0.99 (0.04)	1.00 (0.03)	1.02 (0.01)	1.02 (0.01)	**1.03 (0.00)**
crossintray	2.03 (0.05)	1.96 (0.00)	1.77 (0.17)	2.04 (0.05)	2.00 (0.09)	2.06 (0.05)	2.04 (0.05)	**2.08 (0.05)**
damavandi	-2.90 (0.95)	-9.26 (0.00)	**-2.05 (0.39)**	-2.36 (0.39)	-2.85 (0.88)	-2.16 (0.16)	-2.25 (0.31)	-2.09 (0.09)
dropwave	0.80 (0.11)	0.14 (0.00)	0.65 (0.27)	0.83 (0.10)	0.78 (0.16)	0.81 (0.11)	0.82 (0.10)	**0.83 (0.10)**
easom	**0.10 (0.21)**	0.00 (0.00)	0.09 (0.26)	0.09 (0.16)	0.07 (0.16)	0.07 (0.18)	**0.10 (0.22)**	**0.10 (0.21)**
eggholder	67.89 (11.40)	54.74 (0.00)	28.18 (23.50)	71.59 (10.36)	69.46 (11.04)	70.58 (10.02)	67.77 (10.59)	**74.63 (11.37)**
griewank	-0.20 (0.09)	-1.06 (0.00)	-0.43 (0.27)	-0.18 (0.09)	-0.20 (0.09)	-0.19 (0.09)	-0.19 (0.10)	**-0.17 (0.08)**
himmelblau	-1.44 (1.53)	-3.94 (0.00)	-0.25 (0.51)	-1.30 (1.32)	-1.55 (1.25)	-0.55 (0.60)	-0.50 (0.50)	**-0.20 (0.22)**
holder	16.21 (2.80)	8.83 (0.00)	7.41 (5.49)	18.24 (0.97)	15.48 (3.12)	17.32 (2.22)	17.67 (1.83)	**18.74 (0.52)**
langermann	3.29 (0.60)	**3.98 (0.00)**	1.84 (1.17)	3.26 (0.62)	2.83 (0.88)	3.00 (0.75)	3.23 (0.65)	3.00 (0.93)
levy	-2.44 (2.06)	-0.80 (0.00)	-25.48 (41.98)	-1.27 (1.16)	-2.27 (1.78)	-0.98 (0.71)	-1.49 (1.07)	**-0.47 (0.37)**
michalewicz	1.26 (0.27)	1.36 (0.00)	1.33 (0.45)	1.29 (0.27)	1.24 (0.29)	1.55 (0.24)	1.47 (0.27)	**1.73 (0.10)**
rastrigin	-5.28 (2.57)	-9.63 (0.00)	-11.35 (10.04)	**-3.12 (1.67)**	-5.82 (2.65)	-4.96 (2.56)	-4.82 (2.58)	-4.17 (2.10)
schaffer	**-0.00 (0.00)**	-0.01 (0.00)	-0.01 (0.01)	**-0.00 (0.00)**	**-0.00 (0.00)**	**-0.00 (0.00)**	**-0.00 (0.00)**	**-0.00 (0.00)**
schubert	11.27 (4.58)	1.18 (0.00)	6.59 (5.75)	**11.96 (4.46)**	10.43 (4.43)	9.63 (4.42)	10.39 (4.39)	10.46 (4.84)
colville 4D	-0.15 (0.11)	**-0.06 (0.00)**	-0.12 (0.20)	-0.13 (0.12)	-0.14 (0.11)	-0.13 (0.10)	-0.12 (0.11)	-0.07 (0.06)
hartmann 3D	3.59 (0.19)	3.51 (0.00)	3.70 (0.40)	3.61 (0.18)	3.72 (0.08)	3.80 (0.06)	3.79 (0.06)	**3.84 (0.02)**
hartmann 6D	2.13 (0.45)	0.42 (0.00)	**2.58 (0.39)**	1.97 (0.49)	2.11 (0.48)	2.31 (0.38)	2.33 (0.35)	2.51 (0.32)
rosenbrock 3D	-0.32 (0.15)	-0.54 (0.00)	-0.16 (0.10)	-0.29 (0.15)	-0.20 (0.08)	-0.27 (0.12)	-0.28 (0.13)	**-0.11 (0.04)**
perm 10D	-0.06 (0.07)	-20.96 (0.00)	-0.38 (0.66)	-0.08 (0.08)	-0.05 (0.05)	-0.05 (0.06)	-0.06 (0.06)	**-0.04 (0.04)**
perm 20D	-1.27 (1.08)	nan (nan)	-5.33 (14.95)	-1.49 (1.44)	-1.36 (1.44)	-1.08 (1.02)	-1.30 (1.32)	**-0.88 (0.86)**
# Top-1	2	3	8	3	2	2	3	17

Table 3: Comparison of the maximum returned value using a budget of $n = 100$ for the different algorithms, averaged over 100 repetitions with standard deviations reported. The ECP hyperparameters are fixed across all problems: $\varepsilon_1 = 10^{-2}$, $\tau_{n,d} = \max\left(1 + \frac{1}{nd}, \tau\right)$, with $\tau = 1.001$, and $C = 10^3$. For AdaLIPO, we set $p = 0.1$, as specified in their paper. nan indicates an error related insufficient budget to return a value.

(Bubeck et al., 2011), Zooming Kleinberg et al. (2008), or DOO (Munos, 2011). Among the works relying on an unknown local smoothness, SOO (Munos, 2011; Preux et al., 2014; Kawaguchi et al., 2016) and SequOOL (Bartlett et al., 2019). However, in this work we focus on Lipschitz continious functions, with unkonw Lipschitz constants.

The introduction of the Lipschitz constant, first proposed in the pioneering works of Shubert (1972) and Piyavskii (1972), sparked significant research and has been instrumental in the creation of numerous effective global optimization algorithms, including DIRECT (Jones et al., 1993), MCS (Huyer and Neumaier, 1999), and, more recently, AdaLIPO/AdaLIPO+ (Malherbe and Vayatis, 2017; Serré et al., 2024).

The DIRECT algorithm (Jones et al., 1993) is a Lipschitz optimization algorithm where the Lipschitz constant is unknown. It uses a deterministic splitting technique of the search space in order to sequentially divide and evaluate the function over a subdivision of the space that have recorded the highest upper bound among all subdivisions of similar size for at least a possible value of k. Preux et al. (2014) generalized DIRECT in a broader setting by extending the DOO algorithm to any unknown and arbitrary local semi-metric under the name SOO. The no-regret property of DIRECT was shown in Finkel and Kelley (2004) and Munos et al. (2014) derived convergence rates for SOO using weaker local smoothness assumptions. However, regret upper bounds are not known for SOO or DIRECT.

Malherbe and Vayatis (2017) proposed LIPO algorithm, which relies on the Lipschitz constant to derive acceptance regions, and for an unknown Lipschitz constant, proposed adaptive stochastic strategy which directly relies on the estimation of the Lipschitz constant and presents guarantees for globally Lipschitz functions. It is proven to be a no-regret (consistent) algorithm for Lipschitz functions with unknown Lipschitz constants. AdaLIPO works by sampling and evaluating points uniformly at random with some probability p to estimate the Lipschitz constant k. The estimated constant is then used to identify potentially optimal maximizers based on previously

Objective	PRS	Direct	CMA-ES	DualAnnealing	NeuralUCB	AdaLIPO	AdaLIPO+	ECP (Ours)
ackley	-3.33 (0.91)	-1.64 (0.00)	-10.07 (6.63)	-1.98 (0.94)	-3.04 (0.79)	Infinite Loop	Infinite Loop	**-0.25 (0.11)**
bukin	-10.02 (4.14)	-27.38 (0.00)	**-0.18 (0.17)**	-7.35 (3.61)	-7.58 (4.55)	Infinite Loop	Infinite Loop	-2.86 (1.51)
camel	0.99 (0.06)	1.01 (0.00)	**1.03 (0.00)**	1.02 (0.01)	1.02 (0.01)	Infinite Loop	Infinite Loop	**1.03 (0.00)**
crossintray	2.07 (0.03)	**2.12 (0.00)**	1.72 (0.15)	2.11 (0.02)	2.08 (0.02)	Infinite Loop	Infinite Loop	**2.12 (0.00)**
damavandi	-2.21 (0.18)	-2.01 (0.00)	**-2.00 (0.00)**	-2.05 (0.05)	-2.24 (0.19)	Infinite Loop	Infinite Loop	-2.01 (0.01)
rosenbrock	-0.22 (0.10)	-0.54 (0.00)	**-0.05 (0.00)**	-0.17 (0.05)	-0.12 (0.04)	Infinite Loop	Infinite Loop	-0.08 (0.01)
# Top-1	0	1	4	0	0	0	0	3

Table 4: Comparison of the maximum returned value using a budget of 300 for the different algorithms, averaged over multiple repetitions with standard deviations reported. The ECP hyperparameters are fixed across all problems: $\varepsilon_1 = 10^{-2}$, $\tau_{n,d} = \max\left(1 + \frac{1}{nd}, \tau\right)$, with $\tau = 1.001$, and $C = 10^3$. For AdaLIPO, we set $p = 0.1$, as specified in their paper. Infinite Loop indicates that the method failed to return a valid value within the budget.

explored points, thereby refining the search space. More recently, AdaLIPO+ was introduced as an empirical improvement over AdaLIPO (Serré et al., 2024), which follows the same acceptance process as AdaLIPO, except that the exploration probability p used to estimate k decreases over time. Like our method, both AdaLIPO and AdaLIPO+ optimize the search space with an acceptance condition to evaluate potential maximizers based on the assumed Lipschitzness of the objective function and leveraging previously evaluated points. However, unlike our method, they require additional uniformly random samples from the entire search space to estimate the Lipschitz constant, which makes them less efficient in more extreme budget-constrained scenarios.

Beyond Lipschitz optimization, various global maximization methods have been proposed, each operating under different assumptions and function structures, all aiming to identify a global maximum. One prominent approach in black-box optimization is Bayesian optimization (BO), which builds a probabilistic model of the objective function and uses it to select the most promising points to evaluate (Nogueira, 2014–; Shahriari et al., 2016; Frazier, 2018; Balandat et al., 2020). While several BO algorithms are theoretically guaranteed to converge to the global optimum of the unknown function, they often rely on the assumption that the kernel's hyperparameters are known in advance. To address this limitation, hyperparameter-free approaches such as Adaptive GP-UCB (Berkenkamp et al., 2019) have been proposed. More recently, Lindauer et al. (2022) introduced SMAC3 as a robust and efficient baseline for global optimization. In our empirical evaluation, we demonstrate that ECP outperforms these recent BO baselines.

Evolutionary algorithms, such as CMA-ES (Hansen and Ostermeier, 1996; Hansen, 2006; Hansen et al., 2019), are also known for their practical efficiency, though they do not guarantee a no-regret performance under an infinite evaluation budget (Malherbe and Vayatis, 2017). Simulated annealing (Metropolis et al., 1953; Kirkpatrick et al., 1983), motivated by the physical annealing process when slowly cooling metals, is popular approach for global optimization, later extended to DualAnnealing (Xiang et al., 1997), which combines the generalization of classical simulated annealing and fast simulated annealing (Tsallis, 1988; Tsallis and Stariolo, 1996). However, these methods lack the theoretical regret guarantees for Lipschitz optimization.

Another related class of algorithms is contextual bandits (Auer, 2002; Langford and Zhang, 2007; Filippi et al., 2010; Valko et al., 2013), a special case of reinforcement learning (with a single state) (Sutton, 2018; Haarnoja et al., 2018; Lattimore and Szepesvári, 2020; Fourati et al., 2024a), which involve selecting from a finite set of arms with continuous representations. A notable recent development in this area is NeuralUCB (Zhou et al., 2020), which uses neural networks to estimate upper-confidence bounds for each point. While these methods are primarily designed for selection, they can be adapted for global maximization by randomly sampling points (arms) in each round and selecting the one that maximizes the estimated upper-confidence bound, then retraining the neural network, based on all the previous observation. However, such approaches may be inefficient for small budgets, as neural networks require a large number of samples for effective training.

Finally, while this work focuses on continuous black-box function maximization, other research addresses discrete black-box function maximization, particularly in combinatorial settings. Submodular maximization, a key area within discrete optimization, where recent studies, such as those in (Fourati et al., 2023, 2024c,b), explore effective approximation algorithms for these problems.

Optimization Method	For Lipschitz f	For Unkown k	Stochastic Adaptive	Stopping Guarantees	No-regret	No Space Filling	Finite Budget Regret Bound
PRS	×	✓	×	✓	✓	×	$\mathcal{O}\left(c \cdot \left(\frac{\ln(1/\delta)}{n}\right)^{\frac{1}{d}}\right)$
CMA-ES (Hansen, 2006)	×	✓	✓	✓	×	×	–
DualAnnealing	×	✓	✓	✓	×	–	–
NeuralUCB (Zhou et al., 2020)	×	✓	✓	✓	×	×	–
BayesOpt (Nogueira, 2014–)	×	✓	✓	✓	×	×	–
DIRECT (Jones et al., 1993)	✓	✓	×	✓	–	–	–
SOO (Preux et al., 2014)	×	✓	×	✓	–	–	–
AdaLIPO (Malherbe and Vayatis, 2017)	✓	✓	✓	×	✓	×	$\mathcal{O}\left(c \cdot \left(\frac{5}{p} + \frac{2\ln(\delta/3)}{p\ln\left(1-\Gamma(f,k_{i^\star-1})\right)}\right)^{\frac{1}{d}} \cdot \left(\frac{\ln(3/\delta)}{n}\right)^{\frac{1}{d}}\right)$
AdaLIPO+/AdaLIPO+|ns (Serré et al., 2024)	✓	✓	✓	✓/×	×	×	–
LIPO (Malherbe and Vayatis, 2017)	✓	×	✓	×	✓	✓	$\mathcal{O}\left(c \cdot \left(\frac{\ln(1/\delta)}{n}\right)^{\frac{1}{d}}\right)$
LIPO+/LIPO+|ns (Serré et al., 2024)	✓	×	✓	✓/×	×	✓	–
ECP (ours)	✓	✓	✓	✓	✓	✓	$\mathcal{O}\left(c \cdot \max\left(\left\lceil \log_{\tau_{n,d}}\left(\frac{k}{\varepsilon_1}\right)\right\rceil^{\frac{1}{d}}, 1\right) \cdot \left(\frac{\ln(1/\delta)}{n}\right)^{\frac{1}{d}}\right)$
Lower-Bound	-	-	-	-	-	-	$\Omega(k \cdot \left(\frac{1}{n}\right)^{\frac{1}{d}})$

Table 5: Theoretical Comparison of Global Optimization Methods. Here, $\delta \in (0,1)$ and $c = k \cdot \operatorname{diam}(\mathcal{X})$, where k is the unknown Lipschitz constant. The first column indicates whether the optimization method was proposed for Lipschitz functions. The second column specifies whether the method requires prior knowledge. The third column indicates whether or not the method is stochastic adaptive, stochastically leveraging the collected data and Lipschitzness of the function. The fourth column addresses whether the method is guaranteed to terminate within a specified budget of evaluations. The fifth column indicates whether the method provides no-regret guarantees. The sixth column notes whether the method eliminates blind space filling or pure random search. The final column presents the known finite budget upper bounds on the regret for Lipschitz functions.

C MISSING PROOFS, LEMMAS, AND PROPOSITIONS

C.1 Preliminaries

Lemma 3. *(Malherbe and Vayatis (2017)) If $\mathcal{P}_{k,t}$ denotes the set of potential maximizers of the function f, as defined in Definition 3, then we have $\mathcal{A}_{k,t} = \mathcal{P}_{k,t}$.*

Lemma 4. *(Malherbe and Vayatis (2017)). Let $\mathcal{X} \subset \mathbb{R}^d$ be a compact and convex set with non-empty interior and let $f \in \operatorname{Lip}(k)$ be a k-Lipschitz functions defined on $\mathcal{X}$ for some $k \geq 0$. Then, for any $n \in \mathbb{N}^\star$ and $\delta \in (0,1)$, we have with probability at least $1-\delta$,*

$$\max_{x \in \mathcal{X}} f(x) - \max_{i=1,\cdots,n} f(x_i) \leq k \cdot \operatorname{diam}(\mathcal{X}) \cdot \left(\frac{\ln(1/\delta)}{n}\right)^{\frac{1}{d}}$$

where $x_1, \cdots, x_n$ denotes a sequence of n independent copies of $x \sim \mathcal{U}(\mathcal{X})$.

C.2 Proof of Lemma 1

Assume $\exists x \in \mathcal{A}_{u,t}$, where $\varepsilon_t = u$, we have

$$\min_{i=1,\cdots,t} f(x_i) + u \cdot \|x - x_i\|_2 \geq \max_{i=1,\cdots,t} f(x_i).$$

Hence, for $u \leq v$, then

$$\min_{i=1,\cdots,t} f(x_i) + v \cdot \|x - x_i\|_2 \geq \min_{i=1,\cdots,t} f(x_i) + u \cdot \|x - x_i\|_2 \quad \geq \max_{i=1,\cdots,t} f(x_i).$$

Hence, $x \in \mathcal{A}_{v,t}$. Therefore, $\forall u \leq v, \quad \mathcal{A}_{u,t} \subseteq \mathcal{A}_{v,t}$.

□

C.3 Proof of Proposition 2

Using Lemma 3 from Appendix C.1, we have $\mathcal{P}_{k,t} = \mathcal{A}_{k,t}$. Consider two cases:

- **Case 1**: If $x \in \mathcal{A}_{\varepsilon_t,t}$, then by Lemma 1, for all $\varepsilon_t \leq k$, we have $\mathcal{A}_{\varepsilon_t,t} \subseteq \mathcal{A}_{k,t} = \mathcal{P}_{k,t}$.
- **Case 2**: If $x \in \mathcal{P}_{k,t} = \mathcal{A}_{k,t}$, then by Lemma 1, for all $\varepsilon_t \geq k$, we have $\mathcal{P}_{k,t} = \mathcal{A}_{k,t} \subseteq \mathcal{A}_{\varepsilon_t,t}$.

□

C.4 Proof of Proposition 3

At time $t_2 + 1$, for some value ε, a point $x \in \mathcal{X}$ is accepted if and only if it belongs to a ball $\mathcal{A}_{\varepsilon,t_2+1}$ within $\mathcal{X}$:

$$\min_{1\leq i\leq t_2} f(x_i) + \varepsilon \|x - x_i\|_2 \geq \max_{1\leq i\leq t_2} f(x_i).$$

As $t_1 \leq t_2$, we have

$$\min_{1\leq i\leq t_1} f(x_i) + \varepsilon \|x - x_i\|_2 \geq \min_{1\leq i\leq t_2} f(x_i) + \varepsilon \|x - x_i\|_2.$$

And we have

$$\max_{1\leq i\leq t_2} f(x_i) \geq \max_{1\leq i\leq t_1} f(x_i)$$

Therefore, $\mathcal{A}_{\varepsilon,t_2} \subseteq \mathcal{A}_{\varepsilon,t_1}$. □

C.5 Proof of Proposition 4

At time t, a candidate $x \in \mathcal{X}$ is rejected if and only if it belongs to a ball within $\mathcal{X}$:

$$\min_{1\leq i<t} f(x_i) + \varepsilon_t \|x - x_i\|_2 < \max_{1\leq i<t} f(x_i)$$

Let j be in the arg min of the LHS of the above inequality.

$$\begin{aligned}
f(x_j) + \varepsilon_t \|x - x_j\|_2 < \max_{1\leq i<t} f(x_i) &\iff \varepsilon_t \|x - x_j\| < \max_{1\leq i<t} f(x_i) - f(x_j) \\
&\iff x \in B\left(x_j, \frac{\max_{1\leq i<t} f(x_i) - f(x_j)}{\varepsilon_t}\right) \bigcap \mathcal{X} \\
&\implies x \in B\left(x_j, \frac{\max_{1\leq i<t} f(x_i) - f(x_j)}{\varepsilon_t}\right) \\
&\implies x \in B\left(x_j, \frac{\max_{x\in\mathcal{X}} f(x) - \min_{x\in\mathcal{X}} f(x)}{\varepsilon_t}\right).
\end{aligned}$$

Therefore, the volume of a ball of radius $\frac{\Delta}{\varepsilon_t}$ is an upper bound on the volume that can be removed from the region of potential maximizers, for any sequence of iterations $(x_i)_{1\leq i<t}$. Thus, at time $t+1$, at most the volume of t disjoint balls of radius $\frac{\Delta}{\varepsilon_t}$ could be removed. This leads to the following lower bound on the volume in which potential maximizers should be seek $\mathcal{V}_{t+1}$ verifies:

$$\mathcal{V}_{t+1} \geq \lambda(\mathcal{X}) - \frac{t\pi^{d/2}\Delta^d}{\varepsilon_t^d \Gamma(d/2+1)}$$

As ECP samples candidates uniformly at random in $\mathcal{X}$, the probability of rejecting a candidate is bounded from above by the probability of sampling uniformly at a point in the union of the t disjoint balls.

When ε_t increases within the same iteration t, it is scaled by a multiplicative factor $\tau_{n,d} > 1$ whenever growth is detected, i.e., ε_t becomes $\varepsilon_t \tau_{n,d}^{v_t}$, where v_t represents the number of growth detection within iteration t.

Finally, at time t, by design of the algorithm $\varepsilon_t \geq \varepsilon_1 \tau_{n,d}^{t-1}$. Hence, after v_t growth detection within iteration t, $\varepsilon_t \geq \varepsilon_1 \tau_{n,d}^{t-1} \tau_{n,d}^{v_t}$

□

C.6 AdaLIPO/AdaLIPO+ Rejection Probability

Proposition 7. (AdaLIPO/AdaLIPO+ Rejection Probability) *For any k-Lipschitz function f, let $(x_i)_{1\leq i\leq t}$ be the previously generated and evaluated points of AdaLIPO or AdaLIPO+ until time t. For any $x \in \mathcal{X}$, let $R(x,t)$ be the event of rejecting x at time $t+1$. We have the following upper bound:*

$$\mathbb{P}(R(x,t+1)) \leq \frac{(1-p_t)t(\sqrt{\pi}\Delta)^d}{k_t^d \Gamma(d/2+1)\lambda(\mathcal{X})}$$

where $p_t = \min(1, \ln(\frac{1}{t}))$ and $p_t = p \in (0,1)$, are the exploration probabilities for AdaLIPO+ and AdaLIPO, respectively, to estimate the unknown Lipschitz constant k. $\Delta = \max_{x\in\mathcal{X}} f(x) - \min_{x\in\mathcal{X}} f(x)$, λ is the standard Lebesgue measure that generalizes the notion of volume of any open set, and Γ is the Gamma function given by $\Gamma(x) = \int_0^\infty t^{x-1}e^{-t}\,\mathrm{d}t$.

Proof. We begin by observing that the AdaLIPO+ and AdaLIPO algorithms explore the entire space $\mathcal{X}$ uniformly at random with a probability p_t. Specifically, for AdaLIPO+, the exploration probability is $p_t = \min(1, \ln\left(\frac{1}{t}\right))$, while for AdaLIPO, the exploration probability is a constant $p \in (0,1)$. Both methods aim to estimate the unknown Lipschitz constant k.

Next, note that during exploration, with probability p_t, a point x is accepted with certainty (i.e., rejected with probability zero). To formalize this, we apply the law of total probability to compute the probability of rejecting a point x at time $t+1$:

$$\begin{aligned}
\mathbb{P}(R(x,t+1)) &= (1-p_t)\mathbb{P}(R(x,t+1) \mid \mathcal{E}=0) + p_t\mathbb{P}(R(x,t+1) \mid \mathcal{E}=1) \\
&= (1-p_t)\mathbb{P}(R(x,t+1) \mid \mathcal{E}=0) + p_t \cdot 0 \\
&= (1-p_t)\mathbb{P}(R(x,t+1) \mid \mathcal{E}=0),
\end{aligned}$$

where $\mathcal{E}$ denotes the event that the algorithm is in an exploitation phase (i.e., $\mathcal{E}=1$). The second equality follows from the fact that during exploration ($\mathcal{E}=1$), a point is never rejected, hence $\mathbb{P}(R(x,t+1) \mid \mathcal{E}=0) = 0$. It remains to upper-bound the probability of rejecting a point during the exploitation phase, $\mathbb{P}(R(x,t+1) \mid \mathcal{E}=0)$, which can be done by analyzing the specific rejection criteria of the algorithm during exploitation.

At time t, a candidate $x \in \mathcal{X}$ is rejected if and only if it belongs to a ball within $\mathcal{X}$:

$$\min_{1\leq i<t} f(x_i) + k_t \|x - x_i\|_2 < \max_{1\leq i<t} f(x_i)$$

Let j be in the arg min of the LHS of the above inequality.

$$\begin{aligned}
f(x_j) + \varepsilon_t \|x - x_j\|_2 < \max_{1\leq i<t} f(x_i) &\iff k_t \|x - x_j\| < \max_{1\leq i<t} f(x_i) - f(x_j) \\
&\iff x \in B\left(x_j, \frac{\max_{1\leq i<t} f(x_i) - f(x_j)}{k_t}\right) \bigcap \mathcal{X} \\
&\implies x \in B\left(x_j, \frac{\max_{1\leq i<t} f(x_i) - f(x_j)}{k_t}\right) \\
&\implies x \in B\left(x_j, \frac{\max_{x\in\mathcal{X}} f(x) - \min_{x\in\mathcal{X}} f(x)}{k_t}\right).
\end{aligned}$$

Therefore, the volume of a ball of radius $\frac{\Delta}{k_t}$ is an upper bound on the volume that can be removed from the region of potential maximizers, for any sequence of iterations $(x_i)_{1\leq i<t}$. Thus, at time $t+1$, at most the volume of t disjoint balls of radius $\frac{\Delta}{k_t}$ could be removed. This leads to the following lower bound on the volume in which potential maximizers should be seek $\mathcal{V}_{t+1}$ verifies:

$$\mathcal{V}_{t+1} \geq \lambda(\mathcal{X}) - \frac{t\pi^{d/2}\Delta^d}{k_t^d \Gamma(d/2+1)}$$

As AdaLIPO/AdaLIPO+ samples candidates uniformly at random in $\mathcal{X}$, the probability of rejecting a candidate is bounded from above by the probability of sampling uniformly at a point in the union of the t disjoint balls. □

C.7 Proof of Corollary 1

We define v_t as the number of increases of ε_t at a given iteration t. From Proposition 4 we have

$$\mathbb{P}(R(x,t+1),v_t) \leq \frac{t(\sqrt{\pi}\Delta)^d}{\varepsilon_1^d \tau_{n,d}^{(t-1)d} \tau_{n,d}^{v_t d} \Gamma(d/2+1)\lambda(\mathcal{X})}.$$

Now choose any

$$\begin{aligned} v &\geq \frac{1}{d}\log_{\tau_{n,d}}\left(\frac{2n(\sqrt{\pi}\Delta)^d}{\varepsilon_1^d \Gamma(d/2+1)\lambda(\mathcal{X})}\right) \\ &\geq \frac{1}{d}\log_{\tau_{n,d}}\left(\frac{2t(\sqrt{\pi}\Delta)^d}{\varepsilon_1^d \tau_{n,d}^{(t-1)d} \Gamma(d/2+1)\lambda(\mathcal{X})}\right) \end{aligned}$$

Thus,

$$vd\ln(\tau_{n,d}) \geq \ln\left(\frac{2t(\sqrt{\pi}\Delta)^d}{\varepsilon_1^d \tau_{n,d}^{(t-1)d} \Gamma(d/2+1)\lambda(\mathcal{X})}\right)$$

Then,

$$\tau_{n,d}^{vd} \geq \frac{2t(\sqrt{\pi}\Delta)^d}{\varepsilon_1^d \tau_{n,d}^{(t-1)d} \Gamma(d/2+1)\lambda(\mathcal{X})}$$

Then, we have $\mathbb{P}(R(x,t+1,v)) \leq 1/2$, hence the probability of acceptance $\mathbb{P}(A(x,t+1)) \geq 1/2$. □

C.8 Proof of Proposition 5

We prove this by induction. Notice that $h_1 = 1 \leq 1 \cdot C$. Furthermore, we have $h_{t+1} \leq h_t + C$. Therefore, $h_2 \leq h_1 + C \leq 2 \cdot C$. Now, assume the statement is true for some iteration time $t \geq 1$, i.e., $h_t \leq t \cdot C$. Then,

$$h_{t+1} \leq h_t + C \leq t \cdot C + C \ \leq (t+1) \cdot C.$$

Thus, the proposition holds for all $t \geq 1$ by induction. □

C.9 Proof of Theorem 1

Notice that for any budget $n \in \mathbb{N}^\star$, the computational complexity of running ECP is bounded by $\mathcal{O}(n \cdot v \cdot \max_{t=1\cdots n} h_t)$.

By Proposition 5, we have $\forall t \geq 1$, the stochastic growth condition used in ECP, ensure $h_t \leq t \cdot C \leq n \cdot C$.

Moreover, by Corollary 1, we know with at most $v = \left\lceil \frac{1}{d}\log_{\tau_{n,d}}\left(\frac{2n(\sqrt{\pi}\Delta)^d}{\varepsilon_1^d \Gamma(d/2+1)\lambda(\mathcal{X})}\right)\right\rceil$ increases, the acceptance probability is at least $1/2$. Thus, with $\delta_t = \mathbb{P}(R(x,t+1),v) \leq 1/2$, the growth happens again with at most $\delta \leq (1/2)^{tC}$ after v growths, in round t. Hence, by the union bound, the probability of the growth after v growths happens in any of the t rounds, is at most $\frac{1}{2^C} \cdot \frac{1-\frac{1}{2^C}^t}{1-\frac{1}{2^C}} \leq \frac{1}{2^C} \cdot \frac{1}{1-\frac{1}{2^C}} \leq \frac{1}{2^C-1}$.

Therefore, for any $\varepsilon_1 > 0$, any $\tau_{n,d} > 1$, any constant $C > 1$, and any function $f \in \mathcal{F}$, there exists δ, such as with a probability $1-\delta > 0$, the computational complexity of running ECP is bounded by $\mathcal{O}\left(n^2 \cdot \frac{1}{d}\log_{\tau_{n,d}}\left(\frac{2n(\sqrt{\pi}\Delta)^d}{\varepsilon_1^d \Gamma(d/2+1)\lambda(\mathcal{X})}\right) \cdot C\right)$. □

C.10 Proof of Lemma 2

Notice that for all $t \geq 1$, for any choice of $C > 1$, $\varepsilon_1 > 0$, and $\tau_{n,d} > 1$, ε_t grows throughout the iterations when the stochastic growth condition is satisfied and when a point is evaluated. While the growth condition may or may not occur, the latter happens deterministically after every evaluation. Therefore, $\varepsilon_t \geq \varepsilon_1 \tau_{n,d}^{t-1}$. Hence, the result follows from the non-decreasing and diverging geometric growth of $\varepsilon_1 \tau_{n,d}^{t-1}$. □

C.11 Proof of Theorem 2

For any given function f, with some fixed unknown Lipschitz constant k, and for any chosen constants $\varepsilon_1 > 0$, $\tau_{n,d} > 1$, and $C > 1$, as shown in Lemma 2, there exists a constant $L = \left\lceil \log_{\tau_{n,d}} \left(\frac{k}{\varepsilon_1} \right) \right\rceil$, not depending on n, such that for $t \geq L$, we have $t \geq i^\star$. Hence, by Definition 5, for $t \geq L$, ε_t reaches and exceeds k. Therefore, it is guaranteed that as t tends to infinity, ε_t surpasses k. Furthermore, by Proposition 2, we know that for all $\varepsilon_t > k$, $\mathcal{P}_{k,t} \subseteq \mathcal{A}_{\varepsilon_t,t}$. Thus, as t tends to infinity, the search space uniformly recovers all the potential maximizers and beyond.

C.12 Proof of Proposition 6

We proceed by induction. Let $x_1, \cdots, x_{i^\star}$ be a sequence of evaluation points generated by ECP after $i^\star$ iterations, and let $x'_{i^\star}$ be an independent point randomly sampled over $\mathcal{X}$. Consider any $y \geq \max_{i=1,\cdots,i^\star-1} f(x_i)$, and define the corresponding level set $\mathcal{X}_y = \{x \in \mathcal{X} : f(x) \geq y\}$.

Assume, without loss of generality, that $\mu(\mathcal{X}_y) > 0$ (otherwise, $\mathbb{P}(f(x_{i^\star}) \geq y) = 0$, and the result trivially holds).

Now, recall that for all $t \geq 1$, by Definition 4,

$$\begin{aligned}\mathcal{P}_{k,t} &= \left\{ x \in \mathcal{X} : \exists g \in \mathcal{F}_{k,t} \text{ such that } x \in \arg\max_{x \in \mathcal{X}} g(x) \right\} \\ &= \left\{ x \in \mathcal{X} : \min_{i=1,\cdots,t} (f(x_i) + k \cdot \|x - x_i\|_2) \geq \max_{i=1,\cdots,t} f(x_i) \right\},\end{aligned}$$

where the second equality follows from Lemma 3.

Additionally, by Definition 2, we have

$$\mathcal{X}_{\varepsilon_t,t} := \left\{ x \in \mathcal{X} : \min_{i=1,\cdots,t} (f(x_i) + \varepsilon_t \cdot \|x - x_i\|_2) \geq \max_{i=1,\cdots,t} f(x_i) \right\}.$$

For $t = i^\star$, we have $\varepsilon_{i^\star} \geq k$, implying that $\mathcal{X}_{k,i^\star} \subseteq \mathcal{X}_{\varepsilon_{i^\star},i^\star} \subseteq \mathcal{X}$.

Moreover, since $y \geq \max_{i=1,\cdots,i^\star-1} f(x_i)$, if $\mathcal{X}_y$ is non-empty, its elements are potential maximizers. Hence, $\mathcal{X}_y \subseteq \mathcal{X}_{k,i^\star}$. If $\mathcal{X}_y$ is empty, the result holds trivially.

Next, we compute the following probabilities:

$$\mathbb{P}(f(x_{i^\star}) \geq y) = \mathbb{E}\left[\mathbb{I}\{x_{i^\star} \in \mathcal{X}_y\}\right] = \mathbb{E}\left[\frac{\mu(\mathcal{X}_{\varepsilon_{i^\star},i^\star} \cap \mathcal{X}_y)}{\mu(\mathcal{X}_{\varepsilon_{i^\star},i^\star})}\right] \geq \mathbb{E}\left[\frac{\mu(\mathcal{X}_{k,i^\star} \cap \mathcal{X}_y)}{\mu(\mathcal{X})}\right] = \mathbb{E}\left[\frac{\mu(\mathcal{X}_y)}{\mu(\mathcal{X})}\right] = \mathbb{P}(f(x'_{i^\star}) \geq y).$$

Now, suppose the statement holds for some $n \geq i^\star$. Let $x_1, \cdots, x_{n+1}$ be a sequence of evaluation points generated by ECP after $n+1$ iterations, and let $x'_1, \cdots, x'_{n+1}$ be a sequence of $n+1$ independent points sampled over $\mathcal{X}$.

As before, assume $\mu(\mathcal{X}_y) > 0$, and let $\mathcal{A}_{\varepsilon_n,n}$ denote the sampling region of $x_{n+1} \mid x_1, \cdots, x_n$. Then, on the event $\{\max_{i=i^\star,\cdots,n} f(x_i) < y\}$, we have $\mathcal{X}_y \subseteq \mathcal{X}_{k,n} \subseteq \mathcal{A}_{\varepsilon_n,n} \subseteq \mathcal{X}$.

We now compute:

$$\begin{aligned}
\mathbb{P}\left(\max_{i=i^\star,\cdots,n+1} f(x_i) \geq y\right) &= \mathbb{E}\left[\mathbb{I}\left\{\max_{i=i^\star,\cdots,n} f(x_i) \geq y\right\} + \mathbb{I}\left\{\max_{i=i^\star,\cdots,n} f(x_i) < y, x_{n+1} \in \mathcal{X}_y\right\}\right] \\
&= \mathbb{E}\left[\mathbb{I}\left\{\max_{i=i^\star,\cdots,n} f(x_i) \geq y\right\} + \mathbb{I}\left\{\max_{i=i^\star,\cdots,n} f(x_i) < y\right\} \frac{\mu\left(\mathcal{A}_{\varepsilon_n,n} \cap \mathcal{X}_y\right)}{\mu\left(\mathcal{A}_{\varepsilon_n,n}\right)}\right] \\
&\geq \mathbb{E}\left[\mathbb{I}\left\{\max_{i=i^\star,\cdots,n} f(x_i) \geq y\right\} + \mathbb{I}\left\{\max_{i=i^\star,\cdots,n} f(x_i) < y\right\} \frac{\mu\left(\mathcal{X}_{k,n} \cap \mathcal{X}_y\right)}{\mu\left(\mathcal{A}_{\varepsilon_n,n}\right)}\right] \\
&\geq \mathbb{E}\left[\mathbb{I}\left\{\max_{i=i^\star,\cdots,n} f(x_i) \geq y\right\} + \mathbb{I}\left\{\max_{i=i^\star,\cdots,n} f(x_i) < y\right\} \frac{\mu\left(\mathcal{X}_y\right)}{\mu(\mathcal{X})}\right] \\
&\geq \mathbb{E}\left[\mathbb{I}\left\{\max_{i=i^\star,\cdots,n} f(x'_i) \geq y\right\} + \mathbb{I}\left\{\max_{i=i^\star,\cdots,n} f(x'_i) < y\right\} \frac{\mu\left(\mathcal{X}_y\right)}{\mu(\mathcal{X})}\right] \\
&= \mathbb{E}\left[\mathbb{I}\left\{\max_{i=i^\star,\cdots,n} f(x'_i) \geq y\right\} + \mathbb{I}\left\{\max_{i=i^\star,\cdots,n} f(x'_i) < y, x'_{n+1} \in \mathcal{X}_y\right\}\right] \\
&= \mathbb{P}\left(\max_{i=i^\star,\cdots,n+1} f(x'_i) \geq y\right)
\end{aligned}$$

where the third inequality follows from the fact that $x \mapsto \mathbb{I}\{x \geq y\} + \mathbb{I}\{x < y\}\frac{\mu(\mathcal{X}_y)}{\mu(\mathcal{X})}$ is non-decreasing, and the induction hypothesis implies that $\max_{i=i^\star,\cdots,n} f(x_i)$ stochastically dominates $\max_{i=i^\star,\cdots,n} f(x'_i)$.

Thus, by induction, the statement holds for all $n \geq i^\star$, completing the proof.

□

C.13 Proof of Theorem 3

Fix any $\delta \in (0,1)$. Set $i^\star$ as defined in Definition 5. Considering any $n > i^\star$. As the function satisfies $f \in \mathrm{Lip}(k)$ and for all $t \geq i^\star$, $\varepsilon_t \geq k$, as shown in Proposition 6, for $t \geq i^\star$ the algorithm is always faster or equal to a Pure Random Search with $n - i^\star + 1$ i.i.d. copies of $x' \sim \mathcal{U}(\mathcal{X})$, in achieving higher values than what is already achieved, i.e, for $y \geq \max_{i=1,\cdots,i^\star-1} f(x_i)$. Therefore, using the bound of Lemma 4, we obtain that with probability at least $1-\delta$,

$$\begin{aligned}
\mathcal{R}_{\mathrm{ECP},f}(n) &\leq k \cdot \mathrm{diam}(\mathcal{X}) \cdot \left(\frac{\ln(1/\delta)}{n - i^\star + 1}\right)^{\frac{1}{d}} \\
&= k \cdot \mathrm{diam}(\mathcal{X}) \cdot \left(\frac{n}{n - i^\star + 1}\right)^{\frac{1}{d}} \cdot \left(\frac{\ln(1/\delta)}{n}\right)^{\frac{1}{d}} \\
&\leq k \cdot \mathrm{diam}(\mathcal{X}) \cdot (i^\star)^{\frac{1}{d}} \left(\frac{\ln(1/\delta)}{n}\right)^{\frac{1}{d}}
\end{aligned}$$

where the last inequality follows from Lemma 2.

The result is extended to the case where $n \leq i^\star$ by noticing that the bound is superior to $k \cdot \mathrm{diam}(\mathcal{X})$ in that case, and thus trivial. □

D DETAILS OF THE EXPERIMENTS

The implementations of the ECP are open source and the considered objectives are publicly available at https://github.com/fouratifares/ECP.

D.1 Optimization Algorithms

For DIRECT and Dual Annealing, we use the implementations from SciPy (Virtanen et al., 2020), with standard hyperparameters and necessary modifications to adhere to the specified budgets.

For CMA-ES, we use the implementation described in (Nomura and Shibata, 2024), with standard hyperparameters.

For NeuralUCB, we refer to the authors' implementation in (Zhou et al., 2020). We adapt it to the global optimization setting, where, at each step, we randomly sample four arms and use a neural network with a hidden size of 20 to estimate the upper-confidence bound of these arms, evaluating only the highest one.

We adopt the implementation provided in Botorch (Balandat et al., 2020), setting the number of initial points to 20, the acquisition function to log expected improvement, the number of restarts to 5, and the number of raw samples to 20.

For SMAC3, we utilize the implementation provided by Lindauer et al. (2022), with the default hyperparameters.

For the A-GP-UCB (Berkenkamp et al., 2019), the kernel used is "Matern". The tolerance is set to 1×10^{-2}, and the exploration-exploitation tradeoff is controlled by a gamma value of 10. The noise variance is set to 1×10^{-4}. The initial number of samples is 10, with the starting number of hyperparameters set to 5.

For AdaLIPO and AdaLIPO+, we use the implementation provided by (Serré et al., 2024). To ensure fairness, we run AdaLIPO+ without stopping, thereby maintaining the same budget across all methods. Furthermore, since AdaLIPO requires an exploration probability p, we fix it at 0.1, as done by the authors (Malherbe and Vayatis, 2017).

D.2 Optimization Objectives

We evaluate the proposed method on various global optimization problems using both synthetic and real-world datasets. The synthetic functions were designed to challenge global optimization methods due to their highly non-convex curvatures (Molga and Smutnicki, 2005; Surjanovic and Bingham, 2013), including Ackley, Bukin, Camel, Colville, Cross-in-Tray, Damavandi, Drop-Wave, Easom, Eggholder, Griewank, Hartmann3, Hartmann6, Himmelblau, Holder, Langermann, Levy, Michalewicz, Perm10, Perm20, Rastrigin, Rosenbrock, Schaffer, and Schubert. Some of the 2D functions are shown in Fig. 1 for reference.

For the real-world datasets, we follow the same set of global optimization problems considered in (Malherbe et al., 2016; Malherbe and Vayatis, 2017), drawn from (Frank, 2010). These include Auto-MPG, Breast Cancer Wisconsin, Concrete Slump Test, Housing, and Yacht Hydrodynamics. The task involves optimizing the logarithm of the regularization parameter $\ln(\lambda) \in [-1, 1]$ and the logarithm of the bandwidth $\ln(\sigma) \in [-1, 1]$ of a Gaussian kernel ridge regression by minimizing the empirical mean squared error of the predictions over a 3-fold cross-validation.

D.3 Comparison Protocol

We use the same hyperparameters across all optimization tasks without fine-tuning them for each task, as this may not be practical when dealing with expensive functions and a limited budget.

We allocate a fixed budget of function evaluations, denoted by n, for all methods. The maximum value over the n iterations is recorded for each algorithm. This maximum is then averaged over 100 repetitions, and both the mean and standard deviation are reported.

More evaluations increase the likelihood of finding better points. To ensure that all methods fully utilize the budget, we eliminate any unnecessary stopping conditions, such as waiting times. For example, in AdaLIPO+, we use the variant AdaLIPO+(ns), which continues running even when large rejections occur.

D.4 Compute and Implementations

We implement our method using open source libraries, Python 3.9 and Numpy 1.23.4. We use a CPU 11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80GHz 1.69 GHz. with 16.0 GB RAM

E ECP HYPER-PARAMETER DISCUSSION AND ABLATION STUDIES

E.1 Discussion of ECP Hyperparameters

ECP requires three hyperparameters: $\varepsilon_1 > 0$ (arbitraly small), $\tau_{n,d} > 1$, and $C > 1$. These parameters have been theoretically studied and empirically verified. In our experiments, across all optimization problems represented in Table 1, Table 2, and Table 3, we fixed $\varepsilon_1 = 10^{-2}$ and used $\tau_{n,d} = \max\left(1 + \frac{1}{nd}, \tau\right)$ with $\tau = 1.001$ and $C = 10^3$. Ablation studies have conducted in Appendix E.4 and Appendix E.3.

Increasing the values of ε_1 and $\tau_{n,d}$ causes the rejection probability to approach zero, as shown in Proposition 4, reducing the algorithm to a pure random search and undermining the efficiency of function evaluations. Therefore, smaller values for both ε_1 and $\tau_{n,d}$ are required. By multiplying ε_t by $\tau_{n,d} > 1$ during rejection growth, we guarantee the eventual acceptance of a point, even with small values of $\tau_{n,d}$ and ε_1, as shown in Corollary 1.

Note that a larger constant C implies less constraint on rejection growth, which leads to greater patience before increasing ε_t. Consequently, increasing C results in higher rejection rates, further drifting the algorithm away from pure random search at the cost of potentially longer waiting times to accept a sampled point.

Therefore, either increasing C, decreasing $\tau_{n,d}$, or decreasing ε_1 result in higher rejection rates, which result in more careful acceptance at the cost of an increasing computational complexity of the algorithm, see Theorem 1.

We could have achieved better results with a larger C and smaller ε_1 or τ. However, we fixed these parameters because they demonstrate outstanding performance while still being a fast algorithm. Users of this algorithm can indeed try other values depending on their problem constraints.

E.2 Ablation Study on the Constant ε_1

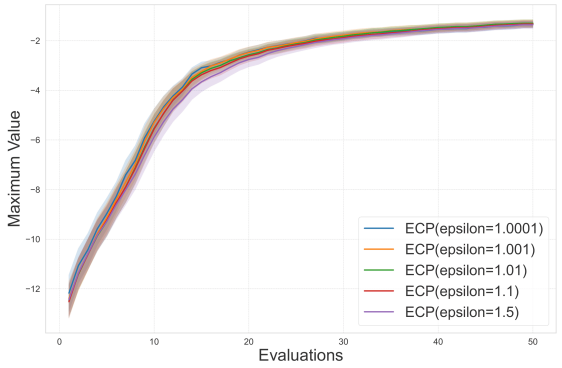
In the following, we test the performance of ECP with various values of ε_1, while keeping $\tau = 10^{-3}$ and $C = 10^3$ fixed. As shown in Figure 4, for different values of $\varepsilon_1 \in [1.0001, 1.001, 1.01, 1.1, 1.5]$, the performance of ECP remains consistent. While for the Hartman 6D function, smaller value of ε_1 lead to noticeably better results, in general, for all functions, smaller ε_1 values lead to better results as predicted by theory. However, in most of the examples, the differences are less significant. Therefore, the performance of ECP is both consistent and robust across different values of ε_1. In our work, we chose a middle value of $\varepsilon_1 = 10^{-2}$, as it achieves good performance while being computationally less expensive than much smaller values of ε_1, as predicted by Theorem 1.

E.3 Ablation Study on the Coefficient τ

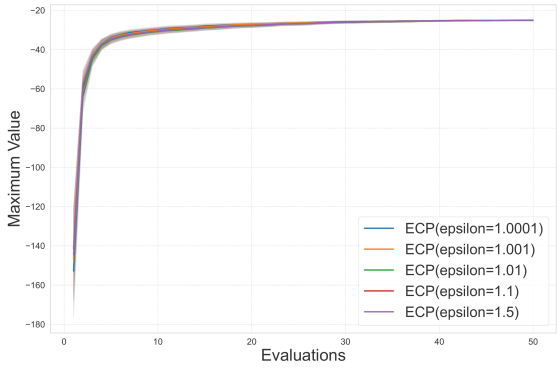
Recall that $\tau_{n,d} = \max\left(1 + \frac{1}{nd}, \tau\right)$, therefore the choice of τ only impacts the algorithm, when the value of τ is larger than $1 + \frac{1}{nd}$. In the following, we test the performance of ECP, with various values of τ, with fixed $C = 10^3$ and $\varepsilon_1 = 10^{-2}$. As shown in Figure 5, for different values of $\tau \in [1.001, 1.01, 1.1, 1.2, 1.4, 1.6, 1.8, 2]$, the performance of ECP remains consistent. While smaller values of τ (blue and orange) yield significantly better results for the Ackley and Hartmann 6D functions, in general, smaller τ values tend to perform better across all functions, as predicted by theory. However, in some cases, such as AutoMPG and Damavandi, the differences are less pronounced. Therefore, the performance of ECP is both consistent and robust across various values of τ. In this work, we chose $\tau = 1.001$, as smaller values would not be considered since $\tau_{n,d} = \max\left(1 + \frac{1}{nd}, \tau\right)$, and we are considering settings with limited budgets.

E.4 Ablation Study on the Constant C

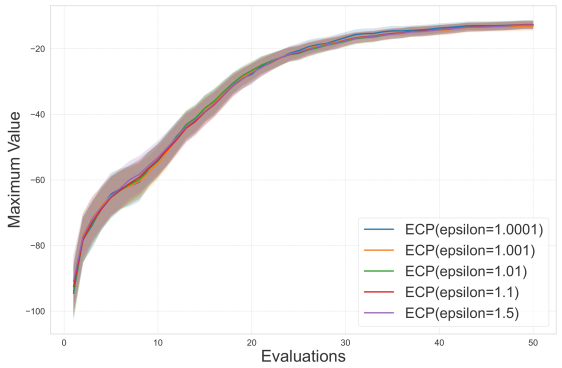
In the following, we test the performance of ECP with various values of C, while keeping $\tau = 10^{-3}$ and $\varepsilon_1 = 10^{-2}$ fixed. As shown in Figure 6, for different values of $C \in [1, 10, 100, 1000, 2000, 4000, 6000, 8000]$, the performance of ECP remains consistent. While for the Ackley function, larger values of C lead to remarkably better results, in general, for all functions, larger C values lead to better results as predicted by theory. However, in examples such as AutoMPG, Damavandi, and Bukin, the differences are less significant. Therefore, the performance of ECP is both consistent and robust across different values of C. In our work, we chose a middle value of $C = 1000$, as it achieves good performance while being computationally less expensive than larger values of C, as predicted by Theorem 1.



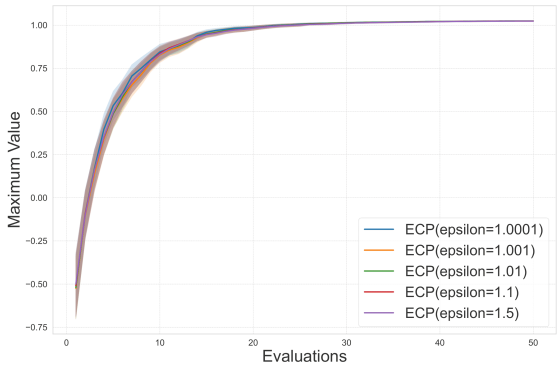
(a) Ackley



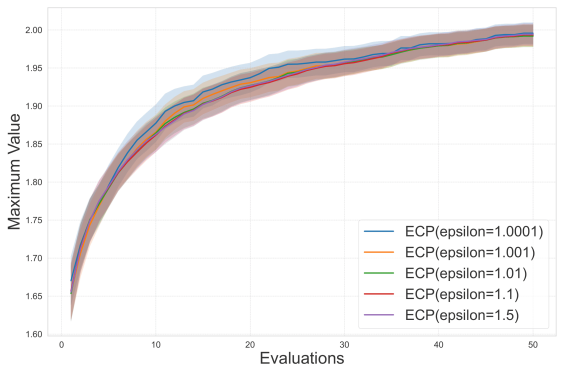
(b) AutoMPG (real-world)



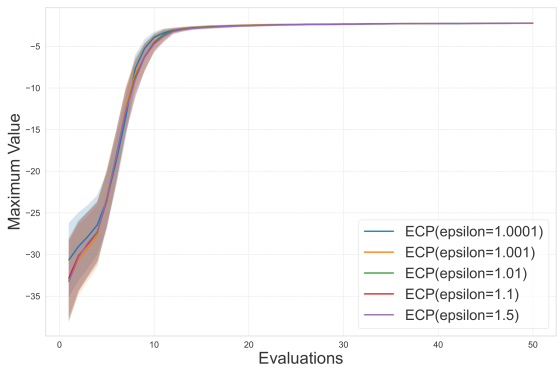
(c) Bukin 2D



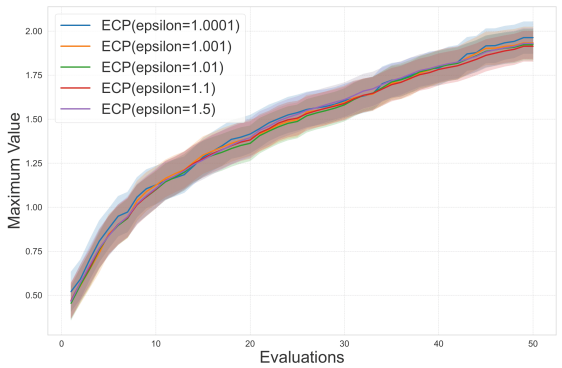
(d) Camel 2D



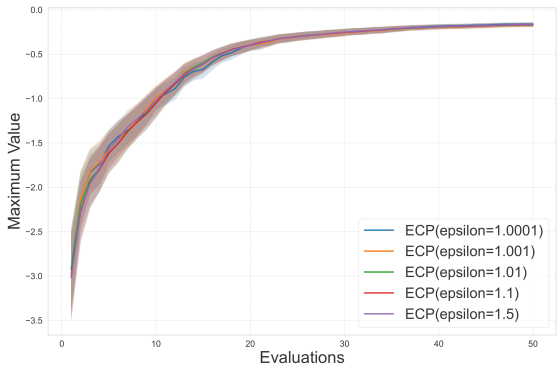
(e) Crossintray 2D



(f) Damavandi 2D

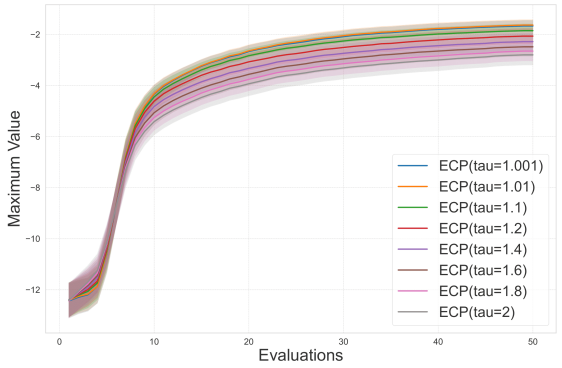


(g) Hartmann 6D

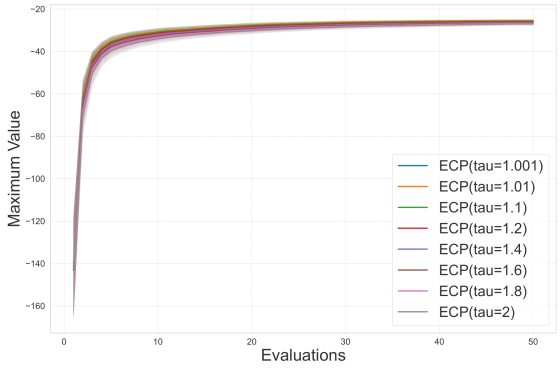


(h) Rosenbrock 3D

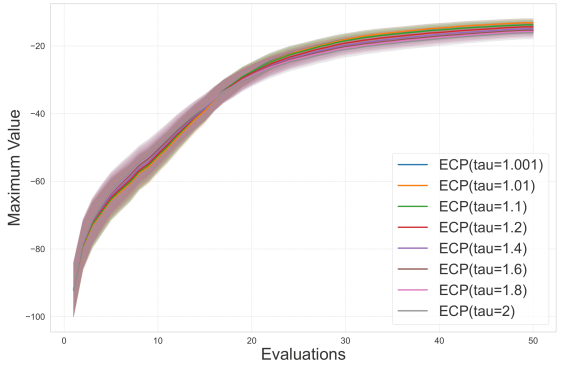
Figure 4: Ablation Study on the Constant $\varepsilon_1 > 0$ of ECP with fixed $C = 10^3$ and $\tau = 10^{-3}$ on various real-world and synthetic non-convex multi-dimensional optimization problems.



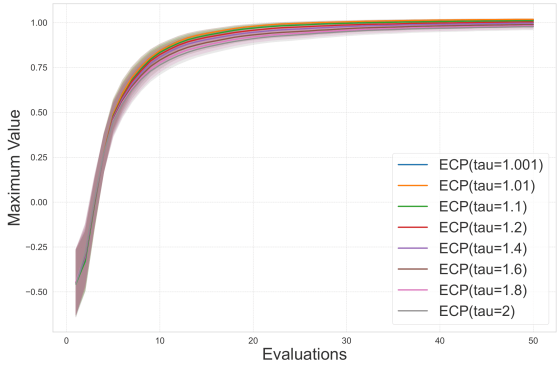
(a) Ackley



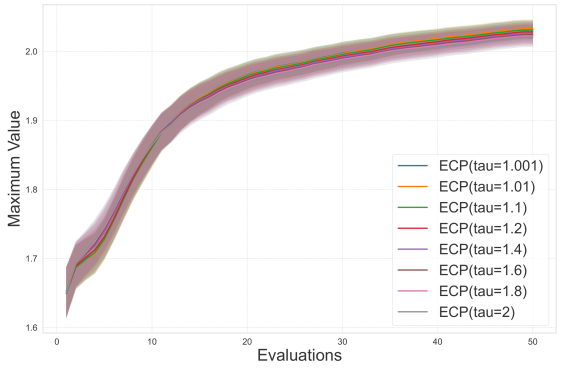
(b) AutoMPG (real-world)



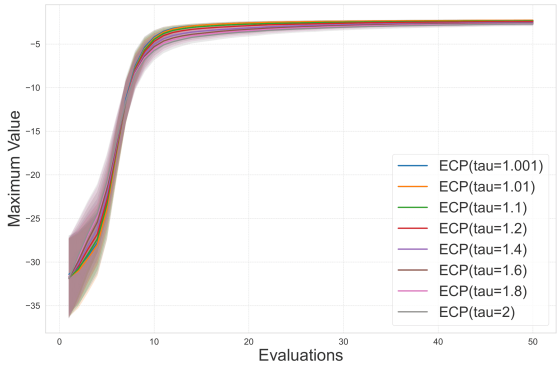
(c) Bukin 2D



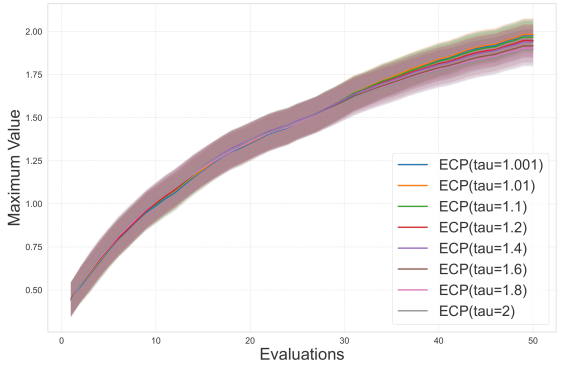
(d) Camel 2D



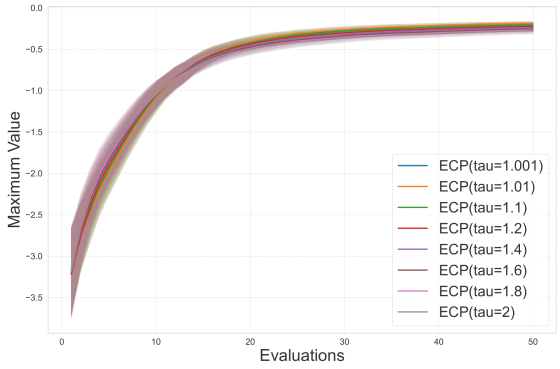
(e) Crossintray 2D



(f) Damavandi 2D

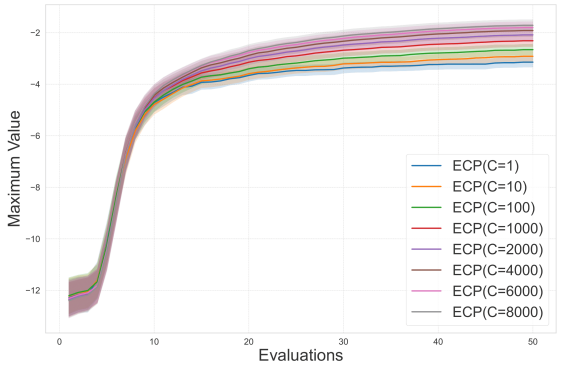


(g) Hartmann 6D

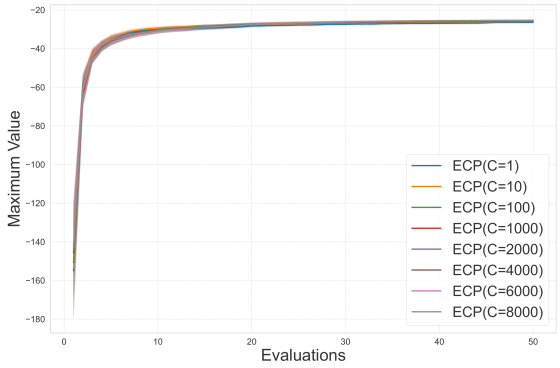


(h) Rosenbrock 3D

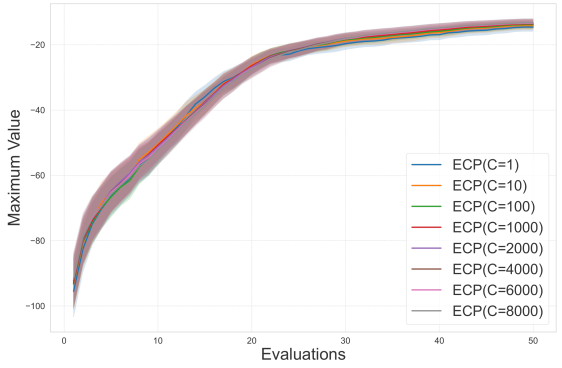
Figure 5: Ablation Study on the Constant $\tau > 1$ of ECP with fixed $C = 10^3$ and $\varepsilon_1 = 10^{-2}$ on various real-world and synthetic non-convex multi-dimensional optimization problems.



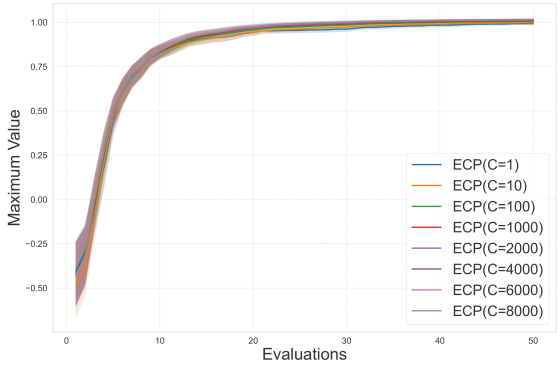
(a) Ackley



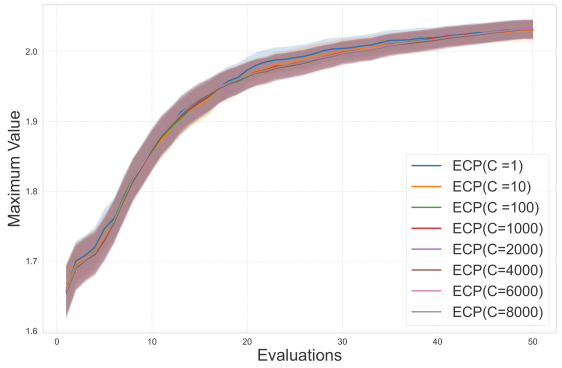
(b) AutoMPG (real-world)



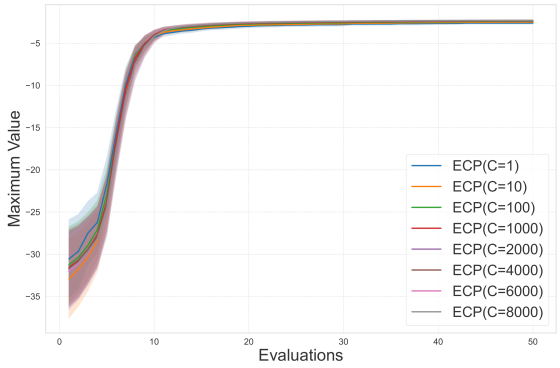
(c) Bukin 2D



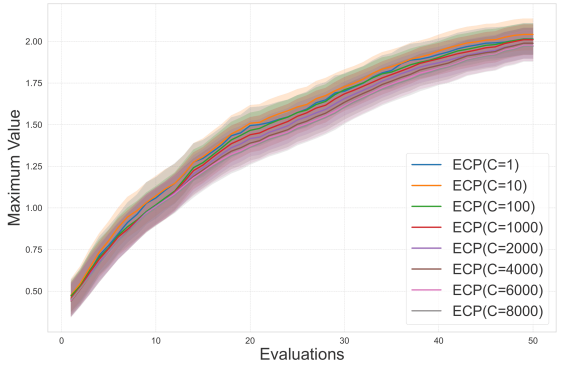
(d) Camel 2D



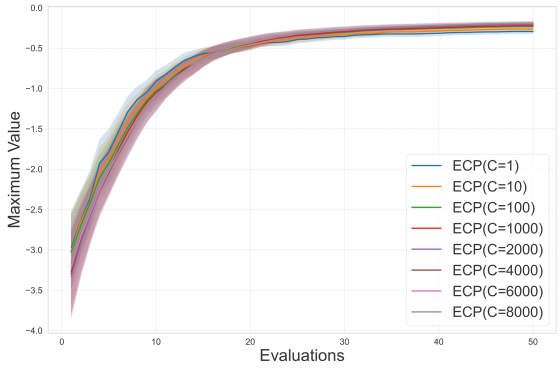
(e) Crossintray 2D



(f) Damavandi 2D



(g) Hartmann 6D



(h) Rosenbrock 3D

Figure 6: Ablation Study on the constant $C > 1$ of ECP with fixed $\tau = 10^{-3}$ and $\varepsilon_1 = 10^{-2}$ on various real-world and synthetic non-convex multi-dimensional optimization problems.