Machine Learning (2020) 109:79–102
https://doi.org/10.1007/s10994-019-05833-y

Combining Bayesian optimization and Lipschitz optimization

Mohamed Osama Ahmed[1] · **Sharan Vaswani**[1] · **Mark Schmidt**[1]

Received: 21 January 2019 / Revised: 13 June 2019 / Accepted: 20 July 2019 / Published online: 22 August 2019

Abstract
Bayesian optimization and Lipschitz optimization have developed alternative techniques for optimizing black-box functions. They each exploit a different form of prior about the function. In this work, we explore strategies to combine these techniques for better global optimization. In particular, we propose ways to use the Lipschitz continuity assumption within traditional BO algorithms, which we call Lipschitz Bayesian optimization (LBO). This approach does not increase the asymptotic runtime and in some cases drastically improves the performance (while in the worst case the performance is similar). Indeed, in a particular setting, we prove that using the Lipschitz information yields the same or a better bound on the regret compared to using Bayesian optimization on its own. Moreover, we propose a simple heuristics to estimate the Lipschitz constant, and prove that a growing estimate of the Lipschitz constant is in some sense "harmless". Our experiments on 15 datasets with 4 acquisition functions show that in the worst case LBO performs similar to the underlying BO method while in some cases it performs substantially better. Thompson sampling in particular typically saw drastic improvements (as the Lipschitz information corrected for its well-known "over-exploration" pheonemon) and its LBO variant often outperformed other acquisition functions.

Keywords Bayesian optimization · Global optimization · Lipschitz optimzation · Optimization

1 Introduction

Bayesian optimization (BO) has a long history and has been used in a variety of fields (see Shahriari et al. 2016), with recent interest from the machine learning community in the context of automatic hyper-parameter tuning (Snoek et al. 2012; Golovin et al. 2017). BO is

Editors: Karsten Borgwardt, Po-Ling Loh, Evimaria Terzi, and Antti Ukkonen.

✉ Mohamed Osama Ahmed
moahmed@cs.ubc.ca

Sharan Vaswani
sharanv@cs.ubc.ca

Mark Schmidt
schmidtm@cs.ubc.ca

[1] University of British Columbia, Vancouver, BC, Canada

an example of a global black-box optimization algorithm (Hendrix et al. 2010; Jones et al. 1998; Pintér 1996; Rios and Sahinidis 2013) which optimizes an unknown function that may not have nice properties such as convexity. In the typical setting, we assume that we only have access to a black box that evaluates the function and that it is expensive to do these evaluations. The objective is to find a global optimum of the unknown function with the minimum number of function evaluations.

The global optimization of a real-valued function is impossible unless we make assumptions about the structure of the unknown function. Lipschitz continuity assumes that the function can't change arbitrarily fast as we change the inputs. This is one of the weakest assumptions under which optimizing an unknown function is still possible. Lipschitz optimization (Piyavskii 1972; Shubert 1972) (LO) exploits knowledge of a bound on the Lipschitz constant L of the function. This constant L specifically gives a bound on the maximum amount that the function can change (as the parameters change). This bound allows LO to prune the search space in order to locate the optimum. In contrast, Bayesian optimization, makes the assumption that the unknown function belongs to a known model class (typically a class of smooth functions), the most common being a Gaussian process (GP) generated using a Gaussian or Matérn kernel (Stein 2012). We review LO and BO in Sect. 2.

Under their own specific sets of additional assumptions, both BO (Bull 2011, Theorem 5) and LO (Malherbe and Vayatis 2017) can be shown to be exponentially faster than random search strategies. If the underlying function is close to satisfying the stronger BO assumptions, then typically BO is able to optimize functions faster than LO. However, when these assumptions are not reasonable, BO may converge slower than simply trying random values (Li et al. 2016; Ahmed et al. 2016). On the other hand, LO makes minimal assumptions (not even requiring differentiability[1]) and simply prunes away values of the parameters that are not compatible with the Lipschitz condition and thus cannot be solutions. This is useful in speeding up simple algorithms like random search. Given a new function to optimize, it is typically not clear which of these strategies will perform better.

In this paper, we propose to combine BO and LO to exploit the advantages of both methods. We call this *Lipschitz Bayesian Optimization* (LBO). Specifically, in Sect. 3, we design *mixed acquisition functions* that use Lipschitz continuity in conjunction with existing BO algorithms. We also address the issue of providing a "harmless" estimate of the Lipschitz constant (see Sect. 2.3), which is an important practical issue for any LO method. Our experiments (Sect. 4) indicate that in some settings the addition of estimated Lipschitz information leads to a huge improvement over standard BO methods. This is particularly true for Thompson sampling, which often outperforms other standard acquisition functions when augmented with Lipschitz information. This seems to be because the estimated Lipschitz continuity seems to correct for the well-known problem of over-exploration (Shahriari et al. 2014). Further, our experiments indicate that it does not hurt to use the Lipschitz information since even in the worst case it does not change the runtime or the performance of the method.

2 Background

We consider the problem of maximizing a real-valued function f with parameters x over a compact set $\mathcal{X}$. We assume that on iteration t, an algorithm chooses a point $x_t \in \mathcal{X}$ and then receives the corresponding function value $f(x_t)$. Typically, our goal is to find the largest

[1] The absolute value function $f(x) = |x|$ is an example of a simple non-differentiable but Lipschitz-continuous function.

possible $f(x_t)$ across iterations. We describe two approaches for solving this problem, namely BO and LO, in detail below.

2.1 Bayesian optimization

BO methods are typically based on Gaussian processes (GPs), since they have appealing universal consistency properties over compact sets and admit a closed-form posterior distribution (Rasmussen and Williams 2006). BO methods typically assume a smooth GP prior on the unknown function, and use the observed function evaluations to compute a posterior distribution over the possible function values at any point x. At iteration t, given the previously selected points $\{x_1, x_2, \dots x_{t-1}\}$ and their corresponding observations $\mathbf{y}_t = [y_1, y_2, \dots, y_{t-1}]$, the algorithm uses an *acquisition function* (based on the GP posterior) to select the next point to evaluate. The value of the acquisition function at a point characterizes the importance of evaluating that point in order to maximize f. To determine x_t, we usually maximize this acquisition function over all x using an auxiliary optimization procedure [(typically we can only approximately solve this maximization (Wilson et al. 2018; Kim and Choi 2019)].

We now formalize the above high-level procedure. We assume that f follows a $GP(0, k(x, x'))$ distribution where $k(x, x')$ is a kernel function which quantifies the similarity between points x and x'. Throughout this paper, we use the Matérn kernel for which $k(x, x') = {\sigma_0}^2 \exp\left(-\sqrt{5}r^2\right)\left(1 + \sqrt{5}r + \frac{5r^2}{3}\right)$ where $r = \sum_{j=1}^{d} \frac{(x_j - x'_j)^2}{\ell_j}$. Here the hyper-parameter ℓ_j is referred to as the length-scale for dimension j and dictates the extent of smoothness we assume about the function f in direction j. The hyper-parameter σ_0 represents the scale of the signal.

We denote the maximum value of the function until iteration t as y_t^* and the set $\{1, 2, \dots, t\}$ as $[t]$. Let $\mathbf{k}_t(x) = [k(x, x_1), k(x, x_2), \dots, k(x, x_t)]$ and let us denote the $t \times t$ kernel matrix as K (so $K_{i,j} = k(x_i, x_j)$ for all $i, j \in [t]$). Given the function evaluations (observations), the posterior distribution at point x after t iterations is given as $\mathcal{P}(f_t(x)) = N(\mu_t(x), {\sigma_t}^2(x))$. Here, the mean and standard deviation of the function at x are given as:

$$\mu_t(x) = \mathbf{k}_t(x)^T \left(K + \sigma^2 I_t\right)^{-1} \mathbf{y}_t,$$
$${\sigma_t}^2(x) = k(x, x) - \mathbf{k}_t(x)^T \left(K + \sigma^2 I_t\right)^{-1} \mathbf{k}_t(x). \tag{1}$$

As alluded to earlier, an acquisition function uses the above posterior distribution in order to select the next point to evaluate the function at. A number of acquisition functions have been proposed in the literature, with the most popular ones: (UCB) (Srinivas et al. 2010), Thompson sampling (TS) (Thompson 1933), expected improvement (EI) (Močkus 1975), probability of improvement (PI) (Kushner 1964), and entropy search (Villemonteix et al. 2009; Hennig and Schuler 2012; Hernández-Lobato et al. 2014). In this work, we focus on four simple widely-used acquisition functions: UCB, TS, EI, and PI. However, we expect that our conclusions would apply to other acquisition functions. For brevity, when defining the acquisition functions, we drop the $(t-1)$ subscripts from $\mu_{t-1}(x)$, $\sigma_{t-1}(x)$, and y_{t-1}^*.
UCB: The acquisition function $UCB(x)$ is defined as:

$$UCB(x) = \mu(x) + \beta_t^{1/2}\sigma(x). \tag{2}$$

Here, β_t is positive parameter that trades off exploration and exploitation.

TS: For TS, in each iteration we first sample a function $\widetilde{f_t}(x)$ from the GP posterior, $\widetilde{f_t} \sim GP(\mu_t(x), \sigma_t(x))$. TS then selects the point x_t which maximizes this deterministic function $\widetilde{f_t}$.

PI: We define the possible improvement (over the current maximum) at x as $I(x) = \max\{f_x(x) - y^*, 0\}$ and the indicator of improvement $u(x)$ as

$$u(x) = \begin{cases} 0, & \text{if } f_x(x) < y^* \\ 1, & \text{if } f_x(x) \geq y^* \end{cases},$$

where $f_x(x) \sim P(f_x)$. PI selects the point x which maximizes the probability of improving over y^*. If $\phi(\cdot)$ and $\Phi(\cdot)$ are the probability density function and the cumulative distribution function for the standard normal distribution, then the PI acquisition function is given as (Kushner 1964):

$$PI(x) = \int_{-\infty}^{\infty} u(x)\phi(f_x(x))\mathrm{d}f_x = \Phi\left(z(x, y^*)\right). \tag{3}$$

where we have defined $z(u, v) = \frac{\mu(u)-v}{\sigma(u)}$.

EI: EI selects an x that maximizes $\mathbb{E}[I(x)]$, where the expectation is over the distribution $\mathcal{P}(f_t(x))$. If $\phi(\cdot)$ is the pdf of the standard normal distribution, the expected improvement acquisition function can be written as (Močkus 1975):

$$\begin{aligned} EI(x) &= \int_{-\infty}^{\infty} I(x)\phi(f_x(x))\mathrm{d}f_x \\ &= \int_{y^*}^{\infty} (f_x(x) - y^*)\phi(f_x(x))\mathrm{d}f_x \\ &= \sigma(x) \cdot \left[z(x, y^*) \cdot \Phi(z(x, y^*)) + \phi(z(x, y^*))\right]. \end{aligned} \tag{4}$$

2.2 Lipschitz optimization

As opposed to assuming that the function comes from a specific family of functions, in LO we simply assume that the function cannot change too quickly as we change x. In particular, we say that a function f is Lipschitz-continuous if for all x and x' we have

$$|f(x) - f(x')| \leq L||x - x'||_2, \tag{5}$$

for a constant L which is referred to as the Lipschitz constant. Note that unlike the typical priors used in BO (like the Gaussian or Matérn kernel), a function can be non-smooth and still be Lipschitz continuous.

Lipschitz optimization methods consider the deterministic (noiseless) case, where $y_i = f(x_i)$. In this setting, Lipschitz optimization uses this Lipschitz inequality in order to test possible locations for the maximum of the function. In particular, at iteration t the Lipschitz inequality implies that the function's value at any x can be upper and lower bounded for any $i \in [t-1]$ by

$$f(x_i) - L||x - x_i||_2 \leq f(x) \leq f(x_i) + L||x - x_i||_2.$$

Since the above inequality holds simultaneously for all $i \in [t-1]$, for any x the function value $f(x)$ can be bounded as:

$$f_{t-1}^l(x) \leq f(x) \leq f_{t-1}^u(x), \text{ where,}$$

$$
\begin{aligned}
f^l_{t-1}(x) &= \max_{i\in[t-1]} \{f(x_i) - L||x - x_i||_2\} \\
f^u_{t-1}(x) &= \min_{i\in[t-1]} \{f(x_i) + L||x - x_i||_2\}
\end{aligned}
\tag{6}
$$

Notice that if $f^u_{t-1}(x) \leq y^*_{t-1}$, then x *cannot* achieve a higher function value than our current maximum y^*_{t-1}.

To exploit these bounds, at each iteration of a typical Lipschitz optimization (LO) method, Malherbe and Vayatis (2017) might sample points x_p uniformly at random from $\mathcal{X}$ until it finds an x_p that satisfies $f^u_{t-1}(x_p) \geq y^*_{t-1}$. If we know the Lipschitz constant L (or use a valid upper bound on the minimum L value), this strategy may prune away large areas of the space while guaranteeing that we do not prune away any optimal solutions. This can substantially decrease the number of function values needed to come close to the global optimum compared to using random points without pruning.

A major drawback of Lipschitz optimization is that in most applications we do not know a valid L. We discuss this scenario in the next section, but first we note that there exist applications where we do have access to a valid L. For example, Bunin and François (2016) discuss cases where L can be dictated by the physical laws of the underlying process (e.g., in heat transfer, solid oxide fuel-cell system, and polymerization). Alternately, if we have a lower and an upper bound on the possible values that the function can take, then we can combine this with the size of $\mathcal{X}$ to obtain an over-estimate of the minimum L value.

2.3 Harmless Lipschitz optimization

When our black-box functions arises from a real world process, a suitable value of L is typically dictated by physical limitations of the process. However, in practice we often do not know L and thus need to estimate it. A simple way to obtain an under-estimate L^{lb}_t of L at iteration t is to use the maximum value that satisfies the Lipschitz inequality across all pairs of points,

$$
L^{lb}_t = \max_{i,j\in[t];x_i\neq x_j} \left\{ \frac{|f(x_i) - f(x_j)|}{||x_i - x_j||_2} \right\}.
\tag{7}
$$

Note that this estimate monotonically increases as we see more examples, but that it may be far smaller than the true L value (and recall that we are considering the noiseless case where $f(x_i) = y_i$). A common variation is to sample several points on a grid (or randomly) to use in the estimate above. Unfortunately, without knowing the Lipschitz constant we do not know how fine this grid should be so in general this may still significantly under-estimate the true quantity.

A reasonable property of any estimate of L that we use is that it is "harmless" in the sense of Ahmed et al. (2016). Specifically, the choice of L should not make the algorithm converge to the global optimum at a slower speed than random guessing (in the worst case). If we have an over-estimate for the minimum possible value of L, then the LO algorithm is harmless as it can only prune values that cannot improve the objective function (although if we over-estimate it by too much then it may not prune much of the space). However, the common under-estimates of L discussed in the previous paragraph are *not* harmless since they may prune the global optima.

We propose a simple solution to the problem that LO is not harmless if we don't have prior knowledge about L: we use a *growing estimate of* L. The danger in using a growing strategy is that if we grow L too slowly then the algorithm may not be harmless. However, in the “Appendix” we show that LO is “harmless” for most reasonable strategies for growing L. This result is not prescriptive in the sense that it does not suggest a practical strategy for growing L (since it depends on the true L), but this result shows that even for enormous values of L that an estimate would have to be growing exceedingly slowly in order for it to not be harmless (exponentially-slow in the minimum value of L, the dimensionality, and the desired accuracy). In our experiments we simply use $L_t^{ub} = \kappa t \cdot L_t^{lb}$, the under-estimator multiplied by the (growing) iteration number and a constant κ (a tunable hyper-parameter). In Sect. 4, we observe that this choice of L_t^{ub} with $\kappa = 10$ consistently works well across 14 datasets with 4 different acquisition functions.

3 Lipschitz Bayesian optimization

In this section, we show how simple changes to the standard acquisition functions used in BO allow us to incorporate the Lipschitz inequality bounds. We call this Lipschitz Bayesian Optimization (LBO). LBO prevents BO from considering values of x^t that cannot be global maxima (assuming we have over-estimated L) and also restricts the range of $f(x_t)$ values considered in the acquisition function to those that are consistent with the Lipschitz inequalities. Figure 1 illustrates the key features of BO, LO, and LBO. It is important to note that the Lipschitz constant L has a different interpretation than the length-scale ℓ of the GP. The constant L specifies an absolute maximum rate of change for the function, while ℓ specifies how quickly a parameterized distance between pairs of points changes the GP. We also note that the computational complexity of using the Lipschitz inequalities is $O(n^2)$ which is the same cost as (exact) inference in the GP (using matrix factorization updates).

We can use the Lipschitz bounds to restrict the limits of the unknown function value for computing the improvement. The upper bound U_f will always be $f^u(x)$, while the lower bound L_f will depend on the relative value of y^*. In particular, we have the following two cases:

$$L_f = \begin{cases} y^*, & \text{if } y^* \in \left(f^l(x), f^u(x)\right) \\ f^u(x), & \text{if } y^* \in (f^u(x), \infty) \end{cases} .$$

The second case represents points that cannot improve over the current best value (that are “rejected” by the Lipschitz inequalities).

Truncated-PI We can define a similar variant for the PI acquisition function as:[2]

$$TPI(x) = \Phi\left(z(x, L_f)\right) - \Phi\left(z(x, U_f)\right). \tag{8}$$

Truncated-EI Using the above bounds, the truncated expected improvement for point x is given by:

$$\begin{aligned} TEI(x) = &- \sigma(x) \cdot z(x, y^*) \left[\Phi(z(x, L_f)) - \Phi(z(x, U_f)\right] \\ &+ \sigma(x) \cdot \left[\phi(z(x, L_f) - \phi(z(x, U_f)\right]. \end{aligned} \tag{9}$$

Note that removing the Lipschitz bounds corresponds to using $f^l(x) = -\infty$ and $f^u(x) = \infty$, and in this case we recover the usual PI and EI methods in Eqs. (3) and (4), respectively.

Truncated-UCB The same strategy can be applied to UCB as follows:

$$TUCB(x) = \min\left\{\mu(x) + \beta_t^{1/2}\sigma(x), f^u(x)\right\}. \tag{10}$$

[2] Note that the only difference between the usual PI/EI and the truncated version is changing the integral limits to (L_f, U_f) instead of $(-\infty, \infty)$.

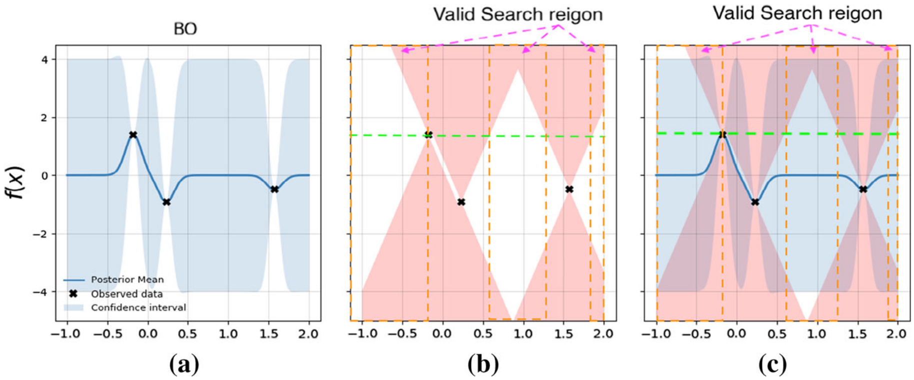


Fig. 1 Visualization of the effect of incorporating the Lipschitz bounds to BO. **a** Shows the posterior mean and confidence interval of the conventional BO. **b** The red color represents the regions of the space that are excluded by the Lipschitz bounds. **c** Shows the effect of LBO. The grey color represents the uncertainty. Using LBO helps cuts off regions where the posterior variance is high, which prevents over-exploration in unnecessary parts of the space (Color figure online)

Accept–Reject An alternative strategy to incorporate the Lipschitz bounds is to use an accept–reject based mixed acquisition function. This approach uses the Lipschitz bounds as a sanity-check to accept or reject the value provided by the original acquisition function, similar to LO methods. Formally, if $g(x)$ is the value of the original acquisition function (e.g. $g(x) = UCB(x)$ or $g(x) = \widetilde{f}(x)$ for TS), then the mixed acquisition function $\overline{g}(x)$ is given as follows:

$$\overline{g}(x) = \begin{cases} g(x), & \text{if } g(x) \in [f^l(x), f^u(x)] \text{ (Accept)} \\ -\infty, & \text{othewise (Reject)} \end{cases}.$$

We refer to the accept–reject based mixed acquisition functions as AR-UCB and AR-TS, respectively. Note that the accept–reject method is quite generic and can be used with any acquisition function that has values on the same scale as that of the function. When using an estimate of L it is possible that a good point could be rejected because the estimate of L is too small, but using a growing estimate ensures that such points can again be selected on later iterations.

3.1 Regret bound for AR-UCB

In this section, we show that under reasonable assumptions, AR-UCB is provably “harmless”, in the sense that it retains the good theoretical properties of GP-UCB. We prove the following theorem under the following assumptions:

1 The GP is correctly specified and with infinite observations, the posterior distribution will collapse to the “true” function f.
2 The noise in the observations σ is small enough for the Lipschitz bounds in Eq. 6 to hold.
3 The Lipschitz constant L is known or has been over-estimated using the techniques described in Sect. 2.3.

Assumption 1 is a common assumption made for providing theoretical results for GP-UCB (Srinivas et al. 2010). Under these assumptions, we obtain the following theorem (proved in “Appendix B”):

Theorem 1 *Let $\mathcal{D}$ be a finite decision space and σ be the standard deviation of the noise in the observations. Let π_t be a positive scalar such that $\sum_t \pi_t^{-1} = 1$ and $\delta \in (0, 1)$. If we use the AR-UCB algorithm with $\beta_t^{1/2} = 2\log(|\mathcal{D}|\pi_t/\delta)$ assuming that the above conditions* 1–3 *hold, then the expected cumulative regret $R(T)$ can be bounded as follows:*

$$R(T) \leq \left(8/\log(1+\sigma^{-2})\right) \beta_T \gamma_T \sqrt{T}.$$

Here, γ_T refers to the information gain for the selected points and depends on the kernel being used. For the squared exponential kernel, we obtain the following specific bound:

$$R(T) \leq \left(8/\log(1+\sigma^{-2})\right) \beta_T (\log(T))^{d+1} \sqrt{T}.$$

The γ_T term can also be bounded for the Matérn kernel following Srinivas et al. (2010). The above theorem shows that under reasonable assumptions, using the Lipschitz bounds in conjunction with GP-UCB cannot result in worse regret. We empirically show that if L is over-estimated, then AR-UCB matches the performance of GP-UCB in the worst case.

Note that the above theorem assumes that the GP is correctly specified with the correct hyper-parameters. It also assumes that we are able to specify the correct value of the trade-off parameter $\beta_t^{1/2}$. These assumptions are not guaranteed to hold in practice and this may result in worse performance of the GP-UCB algorithm. In such cases, our experiments show that using the Lipschitz bounds can lead to better empirical performance than the original GP-UCB.

4 Experiments

Datasets We perform an extensive experimental evaluation and present results on twelve synthetic datasets and three real-world tasks. For the synthetic experiments, we use the standard global-optimization benchmarks namely the Branin, Camel, Goldstein Price, Hartmann (2 variants), Michalwicz (3 variants) and Rosenbrock (4 variants). The closed form and domain for each of these functions is given in Jamil and Yang (2013). As examples of real-world tasks, we consider tuning the parameters for a *robot-pushing* simulation (2 variants) (Wang and Jegelka 2017) and tuning the hyper-parameters for logistic regression (Wu et al. 2017). For the robot pushing example, our aim is to find a good pre-image (Kaelbling and Lozano-Pérez 2017) in order for the robot to push the object to a pre-specified goal location. We follow the experimental protocol from Wang and Jegelka (2017) and use the negative of the distance to the goal location as the black-box function to maximize. We consider tuning the robot position r_x, r_y, and duration of the push t_r for the 3D case. We also tune the angle of the push θ_r to make it a 4 dimensional problem. For the hyper-parameter tuning task, we consider tuning the strength of the ℓ_2 regularization (in the range $[10^{-7}, 0.9]$), the learning rate for stochastic gradient descent (in the range $[10^{-7}, 0.05]$), and the number of passes over the data (in the range [2, 15]). The black-box function is the negative loss on the test set (using a train/test split of 80%/20%) for the MNIST dataset.

Experimental setup For Bayesian optimization, we use a Gaussian Process prior with the Matérn kernel (with a different length scale for each dimension). We modified the publically available BO package *pybo* of Hoffman and Shahriari (2014) to construct the mixed acquisition functions. All the prior hyper-parameters were set and updated across iterations according to the open-source Spearmint package.[3] In order to make the optimization invariant

[3] https://github.com/hips/spearmint.

to the scale of the function values, similar to Spearmint, we standardize the function values; after each iteration, we centre the observed function values by subtracting their mean and dividing by their standard deviation. We then fit a GP to these rescaled function values and correct for our Lipschitz constant estimate by dividing it by the standard deviation. We use DIRECT (Jones et al. 1993) in order to optimize the acquisition function in each iteration. This is one of the standard choices in current works on BO (Eric et al. 2008; Martinez-Cantin et al. 2007; Mahendran et al. 2012), but we expect that Lipschitz information could improve the performance under other choices of the acquisition function optimization approach such as discretization (Snoek et al. 2012), adaptive grids (Bardenet and Kégl 2010), and other gradient-based methods (Hutter et al. 2011; Lizotte et al. 2012). In order to ensure that Bayesian optimization does not get stuck in sub-optimal maxima (either because of the auxiliary optimization or a "bad" set of hyper-parameters), on every fourth iteration of BO (or LBO) we choose a random point to evaluate rather than optimizing the acquisition function. This makes the optimization procedure "harmless" in the sense that BO (or LBO) will not perform worse than random search (Ahmed et al. 2016). This has become common in recent BO methods such as Bull (2011), Hutter et al. (2011), and Falkner et al. (2017), and to make the comparison fair we add this "exploration" step to all methods. Note that in the case of LBO we may need to reject random points until we find one satisfying the Lipschitz inequalities (this does not require evaluating the function). In practice, we found that both the standardization and iterations of random exploration are essential for good performance.[4] All our results are averaged over 10 independent runs, and each of our figures plots the mean and standard deviation of the absolute error (compared to the global optimum) versus the number of function evaluations. For functions evaluated on log scale, we show the 10th and 90th quantiles.

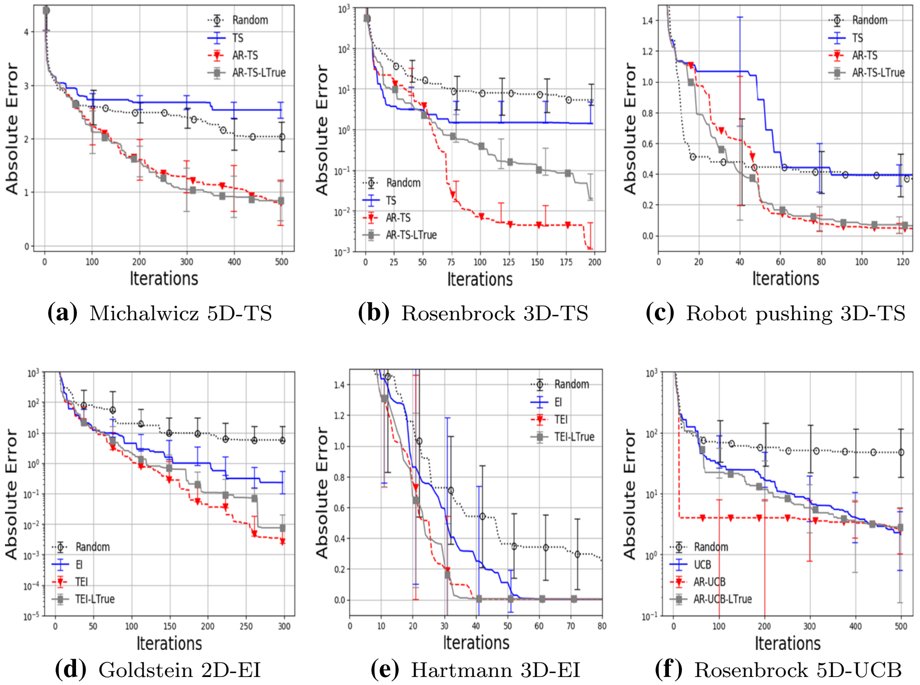
Algorithms compared We compare the performance of Random search, BO, and LBO methods (using both estimated and *True* Lipschitz constant L) for the EI, PI, UCB and TS acquisition functions. The *True* L was estimated offline using a large number of random points. For UCB, we set the trade-off parameter β according to Kandasamy et al. (2017). For EI and PI, we use Lipschitz bounds to truncate the range of function values for calculating the improvement and use the LBO variants TEI and TPI respectively. For UCB and TS, we use the accept–reject strategy and evaluate the LBO variants AR-UCB and AR-TS respectively. In addition to these, we use random exploration as another baseline. We chose the hyper-parameter κ (that controls the extent of over-estimating the Lipschitz constant) on the Rosenbrock-4D function and use the best value of κ for all the other datasets and acquisition functions for both BO and LBO. In particular, we set $\kappa = 10$.

Results To make the results easier to read, we divide the results into the following groups:

1. LBO provides huge improvements over BO shown in Fig. 2. Overall, this represents 21% of all the test cases.
2. LBO provides improvements over BO shown in Fig. 3a. Overall, this represents 9% of all the test cases.
3. LBO performs similar to BO shown in Fig. 3b. Overall, this represents 60% of all the test cases.
4. LBO performs slightly worse than BO shown in Fig. 3c. Overall, this represents 10% of all the test cases.

A comparison of the performance across different acquisition functions (for both BO and LBO) on some of the functions is shown in Fig. 4, where we also show an example of UCB

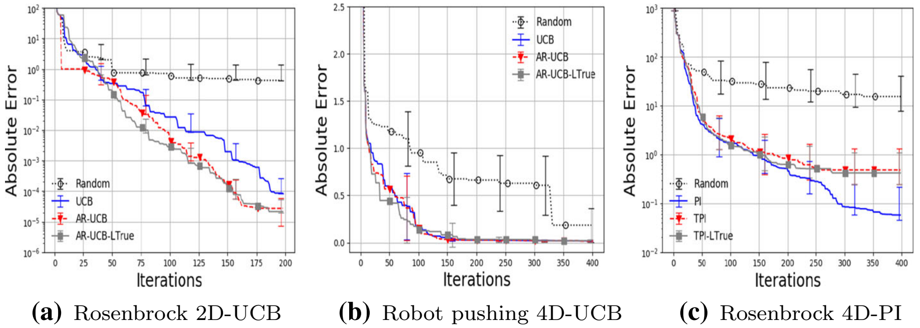
[4] Note that we verified that our baseline version of BO performs better than or equal to Spearmint across benchmark problems.



(a) Michalwicz 5D-TS **(b)** Rosenbrock 3D-TS **(c)** Robot pushing 3D-TS

(d) Goldstein 2D-EI **(e)** Hartmann 3D-EI **(f)** Rosenbrock 5D-UCB

Fig. 2 Examples of functions where LBO provides huge improvements over BO for the different acquisition functions. The figure also shows the performance of random search and LBO using the *True* Lipschitz constant

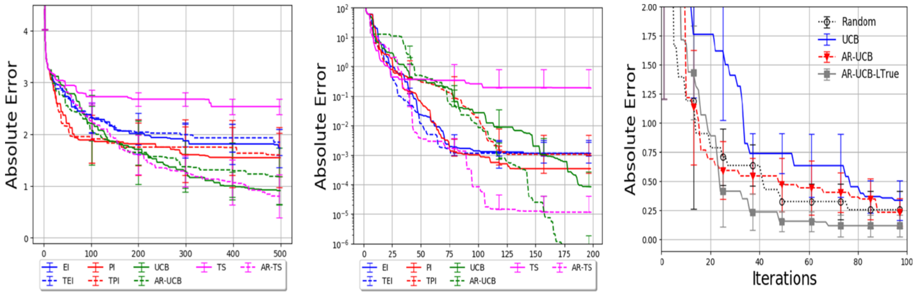


(a) Rosenbrock 2D-UCB **(b)** Robot pushing 4D-UCB **(c)** Rosenbrock 4D-PI

Fig. 3 Examples of functions where LBO provides some improvement over BO (case a), LBO performs similar to BO (case b), and BO performs slightly better than LBO (case c)

where β is misspecified. The plots for all functions and methods are available in "Appendix C". From these experiments, we can observe:

- LBO can potentially lead to large gains in performance across acquisition functions and datasets, particularly for TS.
- Across datasets, we observe that the gains for EI are relatively small, they are occasionally large for PI and UCB and tend to be consistently large for TS. This can be explained as follows: using EI results in under-exploration of the search space, a fact that has been consistently observed and even theoretically proven by Qin et al. (2017). As a result of



(a) Michalwicz 5D **(b)** Rosenbrock 2D **(c)** Camel 2D-UCB

Fig. 4 **a**, **b** Examples of functions where LBO boosts the performance of BO with TS (better seen in color). **c** Example where LBO outperforms BO with UCB when the β parameter is too large ($\beta = 10^{16}$) (Color figure online)

this, BO does not tend to explore "bad" regions when using EI which results in smaller gains from LBO (on the other hand, it may under-explore).

- TS suffers from exactly the opposite problem: it results in high variance leading to over-exploration of the search space and poor performance. This can be observed in Fig. 2a–c where the performance of TS is near random. This has also been observed and noted by Shahriari et al. (2016). For the discrete multi-armed bandit case, Chapelle and Li (2011) multiply the obtained variance estimate by a small number to discourage over-exploration and show that it leads to better results. LBO offers a more principled way of obtaining this same effect and consequently results in making TS more competitive with the other acquisition functions.
- The only functions where LBO slightly hurts are Rosenbrock-4D and Goldstein with UCB and PI.
- For Michalwicz-5D (Fig. 4a), we see that there is no gain for EI, PI, or UCB. However, the gain is huge for TS functions. In fact, even though TS is the worst performing acquisition function on this dataset, its LBO variant AR-TS gives the best performance across all methods. This demonstrates the possible gain that can be obtained from using mixed acquisition functions.
- We observe a similar trend in Fig. 4b where LBO improves TS from near-random performance to being competitive with the best performing methods (while it does not adversely affect the methods performing well).
- For the cases where BO performs slightly better than LBO, we notice that the True estimate of L provides compararble performance to BO, so the problem can be narrowed down to finding a good estimate of L.
- Figure 4c shows examples where LBO saves BO with UCB when the parameter β is chosen too large ($\beta = 10^{16}$). In this case BO performs near random, but using LBO leads to better performance than random search.

In any case, our experiments indicate that LBO methods rarely hurt the performance of the original acquisition function. Since they have minimal computational or memory requirements and are simple to implement, these experiments support using the Lipschitz bounds.

5 Related work

The Lipschitz condition has been used with BO under different contexts in two previous works (González et al. 2016; Sui et al. 2015). The aim of Sui et al. (2015) is to design a "safe" BO algorithm. They assume knowledge of the true Lipschitz constant and exploit Lipschitz continuity to construct a safety threshold in order to construct a "safe" region of the parameter space. This is different than our goal of improving the performance of existing BO methods, and also different in that we estimate the Lipschitz constant as we run the algorithm. On the other hand, González et al. (2016) used Lipschitz continuity to model interactions between a batch of points chosen simultaneously in every iteration of BO (referred to as "Batch" Bayesian optimization). This contrasts with our work where we are aiming to improve the performance of existing sequential algorithms (it is possible that our ideas could be used in their framework).

6 Discussion

In this paper, we have proposed simple ways to combine Lipschitz inequalities with some of the most common BO methods. Our experiments show that this often gives a performance gain, and in the worst case it performs similar to a standard BO method. Although we have focused on four of the simplest acquisition functions, it seems that these inequalities could be used within other acquisition functions. For example, information-theoretic acquisition functions such as entropy search and their recent extensions rely on sampling a function from the GP and hence the techniques we used for Thompson sampling can be used. We leave a systematic study of these information-theoretic acquisition functions to future study. Further, we expect that the Lipschitz inequalities could also be used in other settings like BO with constraints (Gelbart et al. 2014; Hernández-Lobato et al. 2016; Gardner et al. 2014), BO methods based on other model classes like neural networks (Snoek et al. 2015) or random forests (Hutter et al. 2011), and methods that evaluate more than one x_t at a time (Ginsbourger et al. 2010; Wang et al. 2016). Finally, there has been recent interest in first-order Bayesian optimization methods (Ahmed et al. 2016; Wu et al. 2017). If the gradient is Lipschitz continuous then it is possible to use the descent lemma (Bertsekas 2016) to obtain Lipschitz bounds that depend on both function values and gradients.

Acknowledgements We would like to thank the reviewers for their helpful comments. This research was supported by the Canadian Institute for Advanced Research (CIFAR) Learning in Machines & Brains program.

Funding Funding was provided by "CIFAR LMB program".

A Proof for Lipschitz constant estimation

In this section we analyze the minimum number of iterations required before we can guarantee (in expectation) that we'll have a point x satisfying

$$f(x) - f(x^*) \leq \varepsilon, \tag{11}$$

for a given accuracy tolerance ε. Here we assume that x^* is a globally-optimal solution (assumed to exist), the domain of x is a hyper-cube $\mathcal{X}$ in $\mathbb{R}^d$, and f is Lipschitz-continuous. We use L as the minimum value we can use for the Lipschitz constant of f. We first consider the case of random selection, followed by random selection with pruning based on any upper bound on L, and finally random selection with pruning based on a growing estimate of the Lipschitz constant.

A.1 Random selection

Our first result gives a lower bound on the volume of the solution space where the x satisfy (11).

Lemma 1 *For a Lipschitz-continuous function f defined on a hyper-cube $\mathcal{X}$, the volume of $\mathcal{X}$ satisfying (11) is $\Omega((\varepsilon/L)^d)$.*

Proof By the Lipschitz inequality we have for any solution x^* that

$$|f(x) - f(x^*)| \leq L \left\| x - x^* \right\|,$$

for any $x \in \mathcal{X}$. Choose some particular solution x^*, and let $\mathcal{B}$ be the set of x satisfying $L \|x - x^*\| \leq \varepsilon$. Notice that all $x \in \mathcal{B} \cap \mathcal{X}$ satisfy (11), so it is sufficient to show that $|\mathcal{B} \cap \mathcal{X}| = \Omega((\varepsilon/L)^d)$.

Since $\mathcal{B}$ is the set of points satisfying $\|x - x^*\| \leq \varepsilon/L$, it is a hyper-shere of radius ε/L which means its volume is $\frac{\pi^{d/2}(\varepsilon/L)^d}{(d/2)!}$. The case where $\mathcal{B}$ has the smallest intersection with $\mathcal{X}$ is when x^* is at a vertex in the hyper-cube; in this case we have that exactly one orthant of $\mathcal{B}$ intersecting with $\mathcal{X}$. Since there are 2^d orthants (of equal size), in the worst case we have $|\mathcal{B} \cap \mathcal{X}| \geq |\mathcal{B}|/2^d = \frac{\pi^{\mathrm{d}/2}(\varepsilon/L)^d}{2^d(d/2)!} = \Omega((\varepsilon/L)^d)$ (for fixed dimension d). □

Next we give a lower bound on the probability that a random iterate x is a point satisfying (11)

Lemma 2 *For a Lipschitz-continuous function f defined on a hyper-cube $\mathcal{X}$, a point x chosen uniformly at random from $\mathcal{X}$ satisfies (11) with probability $\Omega((\varepsilon/L)^d)$.*

Proof The previous lemma shows that there is a volume of size $\Omega((\varepsilon/L)^d)$ in $\mathcal{X}$ containing solutions. Thus, the probability that random point in $\mathcal{X}$ is a solution is $\Omega((\varepsilon/L)^d/|\mathcal{X}|) = \Omega((\varepsilon/L)^d)$ (for a fixed hyper-cube size). □

Finally, we can give an upper bound on the expected number of iterations before we have an x_t satisfying (11).

Lemma 3 *For a Lipschitz-continuous function f defined on a hyper-cube $\mathcal{X}$, if we independently sample points $\{x_1, x_2, \ldots\}$ uniformly at random from $\mathcal{X}$, then in expectation we find a point x satisfying (11) after $O((L/\varepsilon)^d)$ samples.*

Proof From the previous lemma, each independent sample finds a solution with probability $\Omega((\varepsilon/L)^d$. Viewing each sample as a Bernoulli trial, the expected number of iterations before we find a solution is a geometric random variable with success probability $\Omega((\varepsilon/L)^d$. Since the expectation of a geometric random variable is the inverse of its success probability, in expectation we find a solution after $O((L/\varepsilon)^d)$ samples. □

Instead of "number of samples t to reach accuracy ε", we could equivalently state the result in terms "expected error at iteration t" (simple regret) by inverting the relationship between t and ε. This would give an expected error on iteration t of $O(L/t^{1/d})$.

A.2 Random selection, pruning based on the true Lipschitz constant

In the previous section, we considered choosing points x_t uniformly from $\mathcal{X}$. Consider the case where we are given L (or an upper bound on it), and instead sample uniformly from $\mathcal{X}$ intersected with the points that are not ruled out by the Lipschitz inequalities. Note that this restriction cannot rule out points in $\mathcal{B}$ unless we already have an ε-optimal solution, and thus the arguments from the previous section still apply.

A.3 Random selection, pruning based on a growing Lipschitz constant estimate

Unfortunately, if we use an estimate $\widehat{L}_t$ of L instead of an L satisfying the Lipschitz inequality, we could reject an approximate solution. However, if $\widehat{L}_t$ grows with t then eventually it is sufficiently large that we will not reject an approximate solution (unless we already have an ε-optimal solution). Thus, a crude bound on the expected number of iterations before we find a solution with accuracy ε is given by $O((L/\varepsilon)^d + T)$, where T is the first iteration t beyond which we always have $\widehat{L} \geq L$. Thus, if we choose the sequence $\widehat{L}_t$ such that $T = O((L/\varepsilon)^d)$, then LO with an estimated $\widehat{L}_t$ is harmless as it requires the same expected number of iterations as random guessing. A simple example of a sequence of $\widehat{L}$ values satisfying this property would be to choose $\widehat{L}_t = tL(\varepsilon/L)^d$, which grows extremely-slowly (for small ε and non-trivial d or L). Larger sequences would imply a smaller T and hence also would be harmless.

B Regret bound

Theorem 1 *Let $\mathcal{D}$ be a finite decision space and σ be the standard deviation of the noise in the observations. Let π_t be a positive scalar such that $\sum_t \pi_t^{-1} = 1$ and $\delta \in (0, 1)$. If we use the AR-UCB algorithm with $\beta_t^{1/2} = 2\log(|\mathcal{D}|\pi_t/\delta)$ assuming that the above conditions* 1-3 *hold, then the expected cumulative regret $R(T)$ can be bounded as follows:*

$$R(T) \leq \left(8/\log(1+\sigma^{-2})\right) \beta_T \gamma_T \sqrt{T}.$$

Here, γ_T refers to the information gain for the selected points and depends on the kernel being used. For the squared exponential kernel, we obtain the following specific bound:

$$R(T) \leq \left(8/\log(1+\sigma^{-2})\right) \beta_T (\log(T))^{d+1} \sqrt{T}$$

Proof By definition of Lipschitz bounds and assuming we know the true Lipschitz constant L, at iteration t, for all x,

$$f_{t-1}^l(x) \leq f(x) \leq f_{t-1}^u(x). \tag{12}$$

We now use the following lemma from Srinivas et al. (2010):

Lemma 4 (Lemma 5.1 in Srinivas et al. (2010)) *Denoting $\mathcal{D}$ as a finite decision space, let $\pi_t > 0$ and $\sum_t \pi_t^{-1} = 1$. Choose $\beta_t^{1/2} = 2\log(|\mathcal{D}|\pi_t/\delta)$ where $\delta \in (0, 1)$. Then, for all $x \in \mathcal{D}$ and $t \geq 1$, with probability $1-\delta$,*

$$|f(x) - \mu_{t-1}(x)| \leq \beta_t^{1/2}\sigma_{t-1}(x). \tag{13}$$

From Eqs. (12) and (13),

$$f(x^*) \leq \min\{f_{t-1}^u(x^*), \mu_{t-1}(x^*) + \beta_t^{1/2}\sigma_{t-1}(x^*)\}. \tag{14}$$

For the point x_t selected at round t, the following relation holds because of the accept–reject condition:

$$f_{t-1}^l(x_t) \leq \mu_{t-1}(x_t) + \beta_t^{1/2}\sigma_{t-1}(x_t) \leq f_{t-1}^u(x_t). \tag{15}$$

The following holds because of the definition of the UCB rule:

$$\mu_{t-1}(x_t) + \beta_t^{1/2}\sigma_{t-1}(x_t) \geq \mu_{t-1}(x^*) + \beta_t^{1/2}\sigma_{t-1}(x^*). \tag{16}$$

From Eqs. (13) and (15)

$$\mu_{t-1}(x_t) + \beta_t^{1/2}\sigma_{t-1}(x_t) \leq \min\{f(x_t) + 2\beta_t^{1/2}\sigma_{t-1}(x_t), f_{t-1}^u(x_t)\}. \tag{17}$$

Let r_t be the instantaneous regret in round t. Then,

$$\begin{aligned}
r_t &= f(x^*) - f(x_t) \\
&\leq \min\{f_{t-1}^u(x^*), \mu_{t-1}(x^*) + \beta_t^{1/2}\sigma_{t-1}(x^*)\} - f(x_t) \qquad \text{(From Eq. 14)} \\
&\leq \min\{f_{t-1}^u(x^*), \mu_{t-1}(x_t) + \beta_t^{1/2}\sigma_{t-1}(x_t)\} - f(x_t) \qquad \text{(From Eq. 16)} \\
&= \min\{f_{t-1}^u(x^*) - f(x_t), \mu_{t-1}(x_t) + \beta_t^{1/2}\sigma_{t-1}(x_t) - f(x_t)\} \\
&\qquad\qquad\qquad (\min\{a, b\} - c = \min\{a - c, b - c\}) \\
&\leq \mu_{t-1}(x_t) + \beta_t^{1/2}\sigma_{t-1}(x_t) - f(x_t) \qquad (\min\{a, b\} \leq b) \\
&\leq \min\{f(x_t) + 2\beta_t^{1/2}\sigma_{t-1}(x_t), f_{t-1}^u(x_t)\} - f(x_t) \qquad \text{(From Eq. 17)} \\
&= \min\{2\beta_t^{1/2}\sigma_{t-1}(x_t), f_{t-1}^u(x_t) - f(x_t)\} \qquad (\min\{a, b\} - c = \min\{a - c, b - c\}) \\
\Longrightarrow \quad r_t &\leq \min\{2\beta_t^{1/2}\sigma_{t-1}(x_t), f_{t-1}^u(x_t) - f_{t-1}^l(x_t)\}. \qquad \text{(From Eq. 12)}
\end{aligned}$$

Let us now consider the term $f_{t-1}^u(x_t) - f_{t-1}^l(x_t)$.

$$\begin{aligned}
f_{t-1}^u(x_t) - f_{t-1}^l(x_t) &= \min_{i\in[t-1]}\{f(x_i) + L||x_t - x_i||_2\} - \max_{i\in[t-1]}\{f(x_i) - L||x_t - x_i||_2\} \\
&\quad \text{(By Eq.6)} \\
&= \min_{i\in[t-1]}\{f(x_i) + L||x_t - x_i||_2\} + \min_{i\in[t-1]}\{-f(x_i) + L||x_t - x_i||_2\} \\
&\quad (-\max\{a, b\} = \min\{-a, -b\}) \\
&\leq \min_{i\in[t-1]}\{f(x_i) + L||x_t - x_i||_2 - f(x_i) + L||x_t - x_i||_2\} \\
&\quad (\min\{a_i + b_i\} \geq \min\{a_i\} + \min\{b_i\}) \\
\Longrightarrow \quad f_{t-1}^u(x_t) - f_{t-1}^l(x_t) &\leq 2L \min_{i\in[t-1]}\{||x_t - x_i||_2\}.
\end{aligned}$$

From the above equations,

$$r_t \leq \min\left\{2\beta_t^{1/2}\sigma_{t-1}(x_t), 2L \min_{i\in[t-1]}\{||x_t - x_i||_2\}\right\}.$$

Let $R(T)$ be the cumulative regret after T rounds.

$$R(T) = \sum_{t=1}^{T} r_t \leq \sum_{t=1}^{T} \left[\min \left\{ 2\beta_t^{1/2}\sigma_{t-1}(x_t), 2L \min_{i \in [t-1]} \{||x_t - x_i||_2\} \right\} \right]$$

$$R(T) \leq \min \left\{ 2\sum_{t=1}^{T} \beta_t^{1/2}\sigma_{t-1}(x_t), 2L \sum_{t=1}^{T} \min_{i \in [t-1]} \{||x_t - x_i||_2\} \right\} \qquad (\min\{\sum_i a_i\} \geq \sum_i \min\{a_i\})$$

We now bound the term $2\sum_{t=1}^{T} \beta_t^{1/2}\sigma_{t-1}(x_t)$ using the lemma in Srinivas et al. (2010) which we restate next:

Lemma 5 (Lemma 5.4 in Srinivas et al. (2010)) *Choosing* $\beta_t^{1/2} = 2\log(|\mathcal{D}|\pi_t/\delta)$,

$$2\sum_{t=1}^{T} \beta_t^{1/2}\sigma_{t-1}(x_t) \leq C_1\gamma_T\sqrt{T}.$$

where $C_1 = \left(8/\log(1+\sigma^{-2})\right)\beta_T$. *Here* γ_T *refers to the information gain for the selected points.*

Using the above lemma, we obtain the following bound:

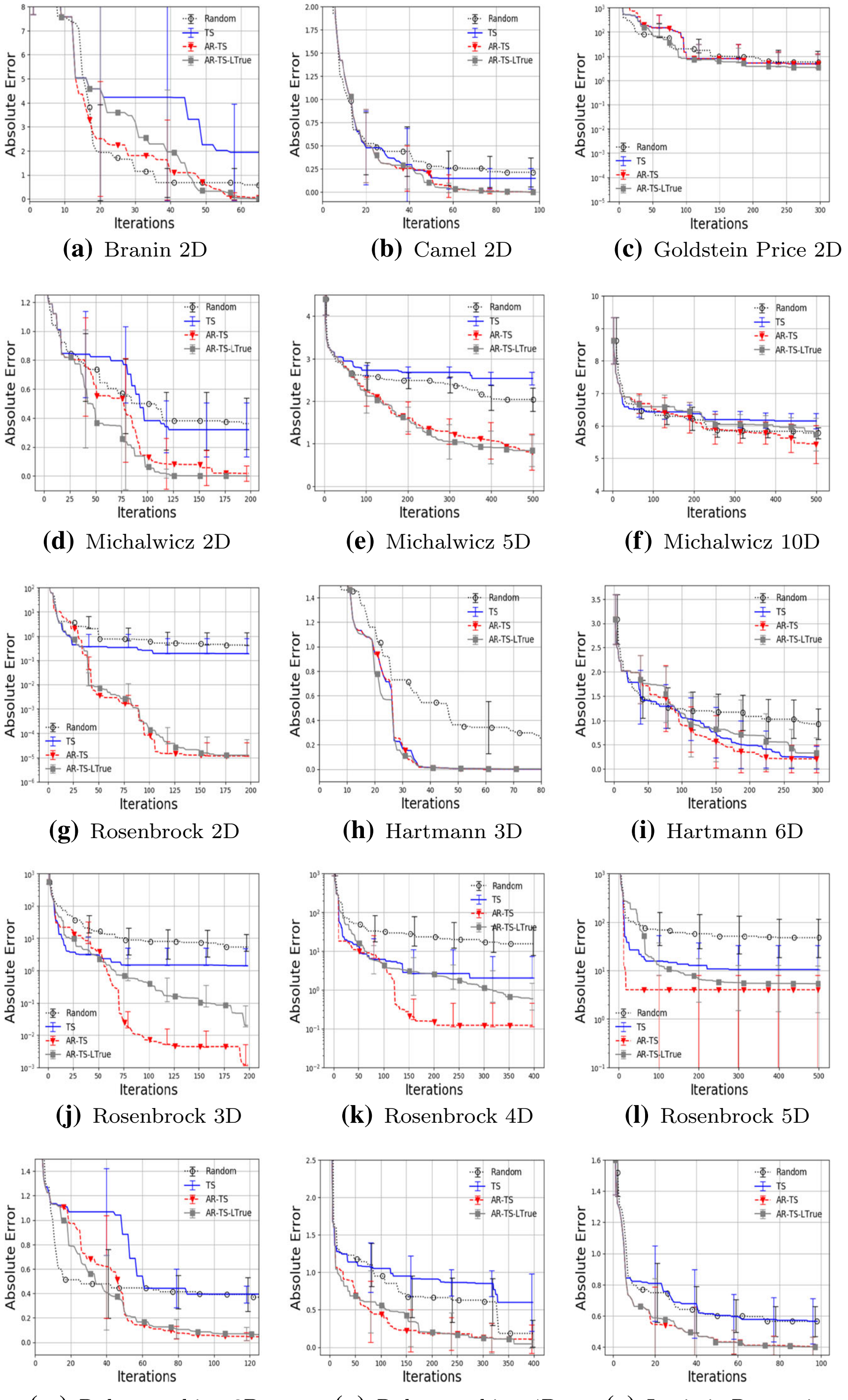
$$R(T) \leq \min \left\{ C_1\gamma_T\sqrt{T}, 2L\sum_{t=1}^{T} \min_{i \in [t-1]} \{||x_t - x_i||_2\} \right\}$$
$$\implies R(T) \leq \left(8/\log(1+\sigma^{-2})\right)\beta_T\gamma_T\sqrt{T}.$$

□

C Additional experimental results

Below we show the results of all the experiments for all the datasets as follows:

- Figure 5 shows the performance of Random search, BO, and LBO (using both estimated and *True* L) for the TS acquisition function.
- Figure 6 shows the performance of Random search, BO, and LBO (using both estimated and *True* L) for the UCB acquisition function.
- Figure 7 shows the performance of Random search, BO, and LBO (using both estimated and *True* L) for the EI acquisition function.
- Figure 8 shows the performance of Random search, BO, and LBO (using both estimated and *True* L) for the PI acquisition function.
- Figure 9 shows the performance of BO and LBO using the estimated L for the all acquisition function.
- Figure 10 shows the performance of Random search, BO, and LBO (using both estimated and *True* L) for the UCB acquisition function with very large $\beta = 10^{16}$.



(a) Branin 2D **(b)** Camel 2D **(c)** Goldstein Price 2D

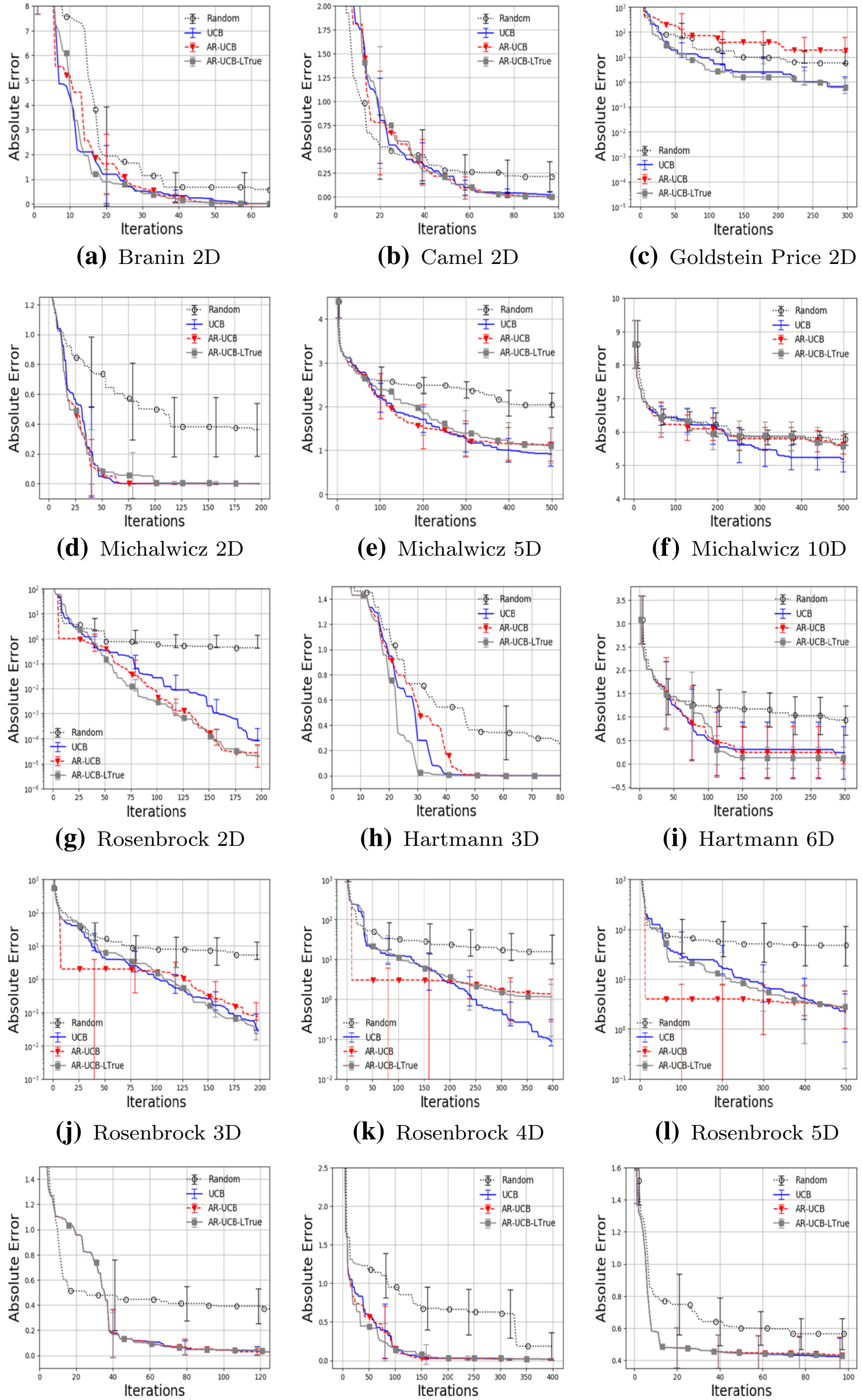
(d) Michalwicz 2D **(e)** Michalwicz 5D **(f)** Michalwicz 10D

(g) Rosenbrock 2D **(h)** Hartmann 3D **(i)** Hartmann 6D

(j) Rosenbrock 3D **(k)** Rosenbrock 4D **(l)** Rosenbrock 5D

(m) Robot pushing 3D **(n)** Robot pushing 4D **(o)** Logistic Regression

Fig. 5 Comparing the performance of the conventional BO acquisition function, corresponding LBO mixed acquisition function, Lipschitz optimization and random exploration for the TS acquisition functions



(a) Branin 2D **(b)** Camel 2D **(c)** Goldstein Price 2D

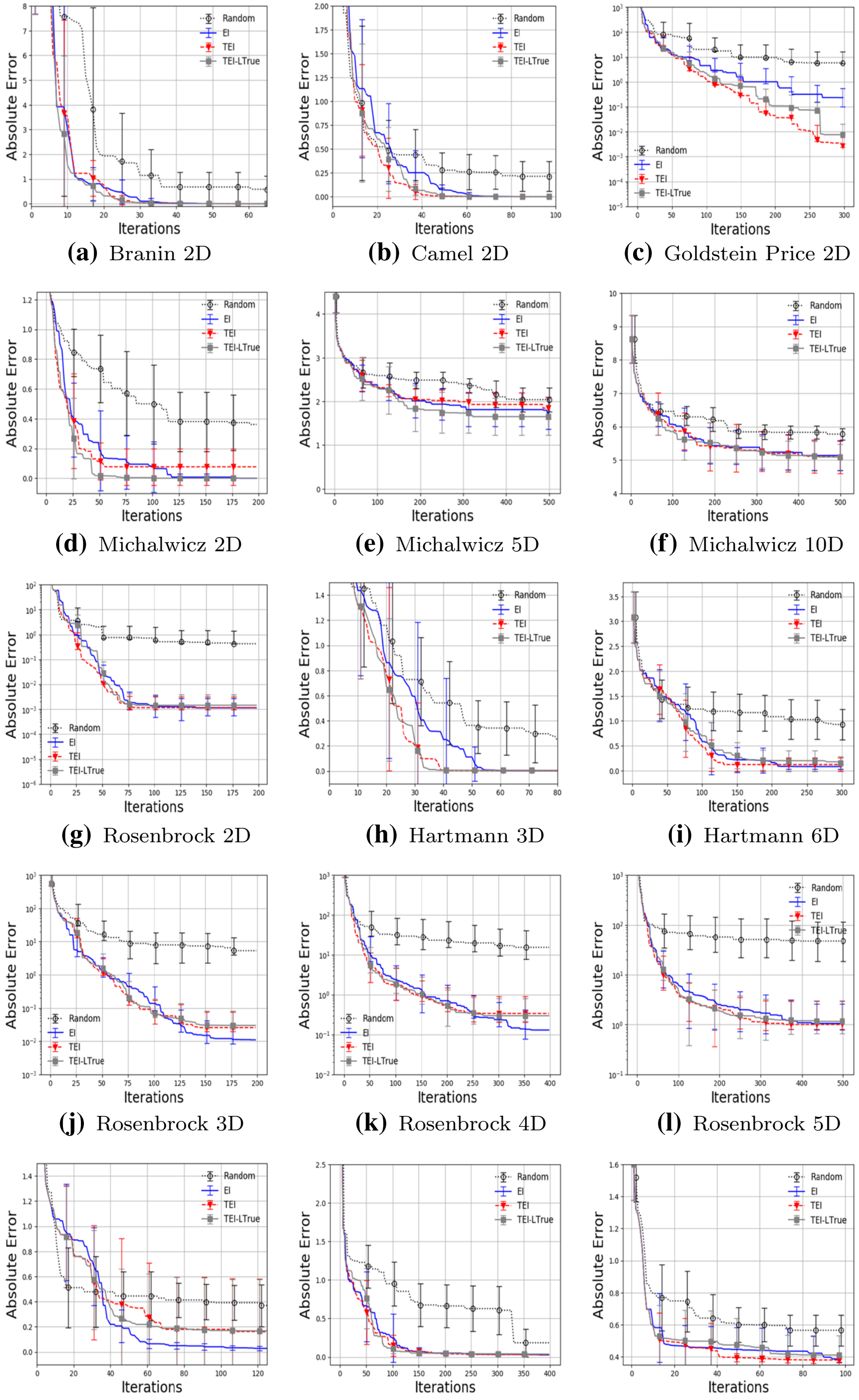
(d) Michalwicz 2D **(e)** Michalwicz 5D **(f)** Michalwicz 10D

(g) Rosenbrock 2D **(h)** Hartmann 3D **(i)** Hartmann 6D

(j) Rosenbrock 3D **(k)** Rosenbrock 4D **(l)** Rosenbrock 5D

(m) Robot pushing 3D **(n)** Robot pushing 4D **(o)** Logistic Regression

Fig. 6 Comparing the performance of the conventional BO acquisition function, corresponding LBO mixed acquisition function, Lipschitz optimization and random exploration for the UCB acquisition functions



(a) Branin 2D **(b)** Camel 2D **(c)** Goldstein Price 2D

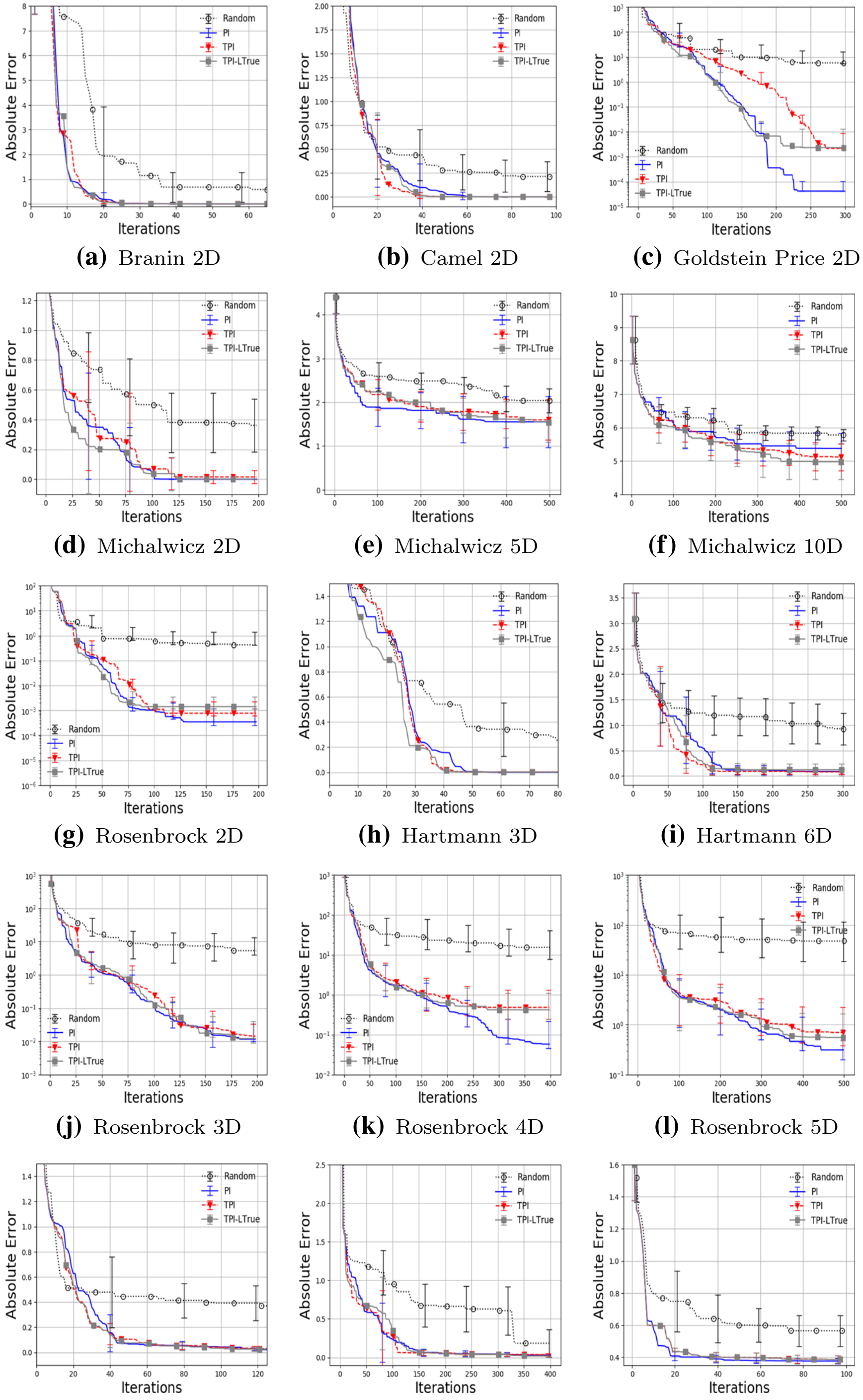
(d) Michalwicz 2D **(e)** Michalwicz 5D **(f)** Michalwicz 10D

(g) Rosenbrock 2D **(h)** Hartmann 3D **(i)** Hartmann 6D

(j) Rosenbrock 3D **(k)** Rosenbrock 4D **(l)** Rosenbrock 5D

(m) Robot pushing 3D **(n)** Robot pushing 4D **(o)** Logistic Regression

Fig. 7 Comparing the performance of the conventional BO acquisition function, corresponding LBO mixed acquisition function, Lipschitz optimization and random exploration for the EI acquisition functions



(a) Branin 2D **(b)** Camel 2D **(c)** Goldstein Price 2D

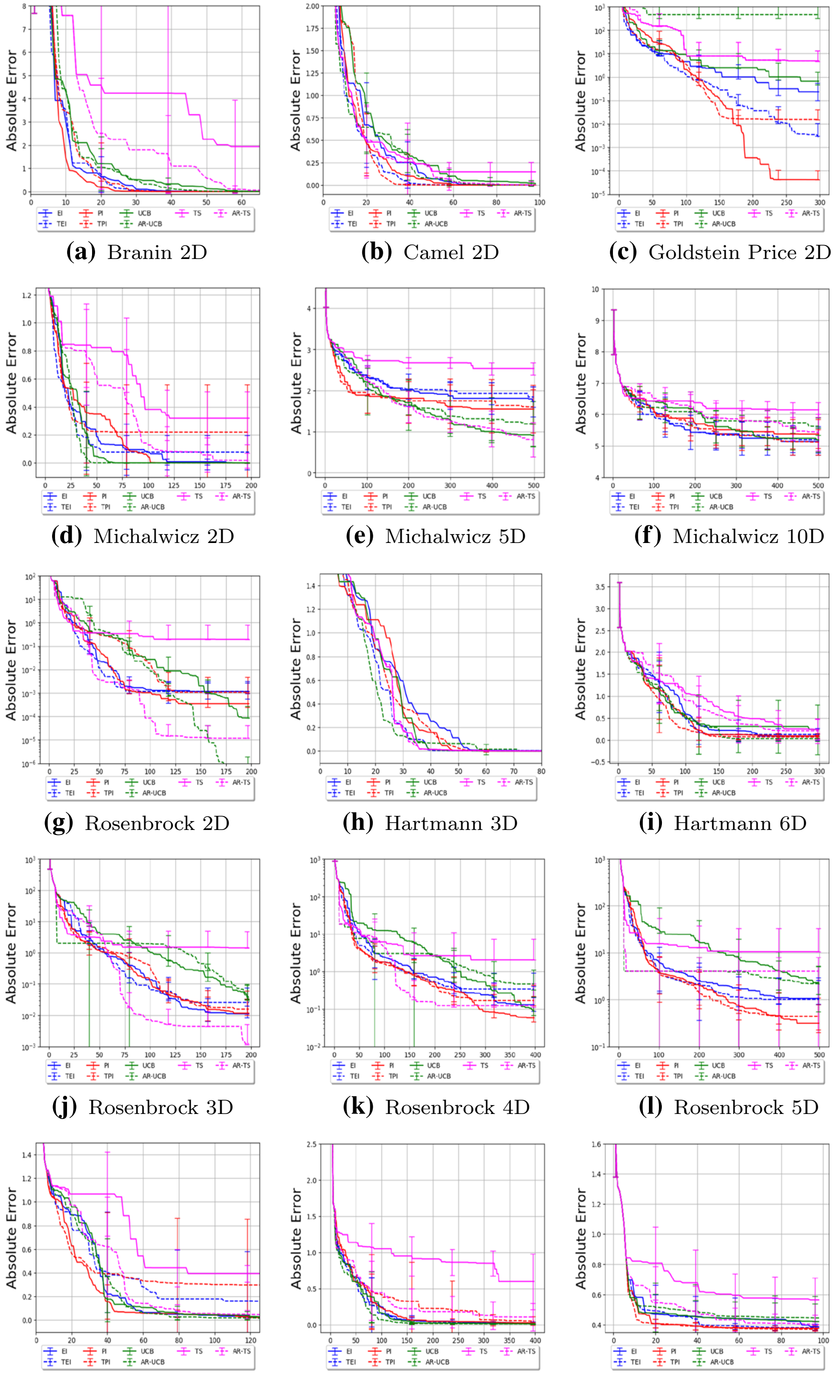
(d) Michalwicz 2D **(e)** Michalwicz 5D **(f)** Michalwicz 10D

(g) Rosenbrock 2D **(h)** Hartmann 3D **(i)** Hartmann 6D

(j) Rosenbrock 3D **(k)** Rosenbrock 4D **(l)** Rosenbrock 5D

(m) Robot pushing 3D **(n)** Robot pushing 4D **(o)** Logistic Regression

Fig. 8 Comparing the performance of the conventional BO acquisition function, corresponding LBO mixed acquisition function, Lipschitz optimization and random exploration for the PI acquisition functions



(a) Branin 2D **(b)** Camel 2D **(c)** Goldstein Price 2D

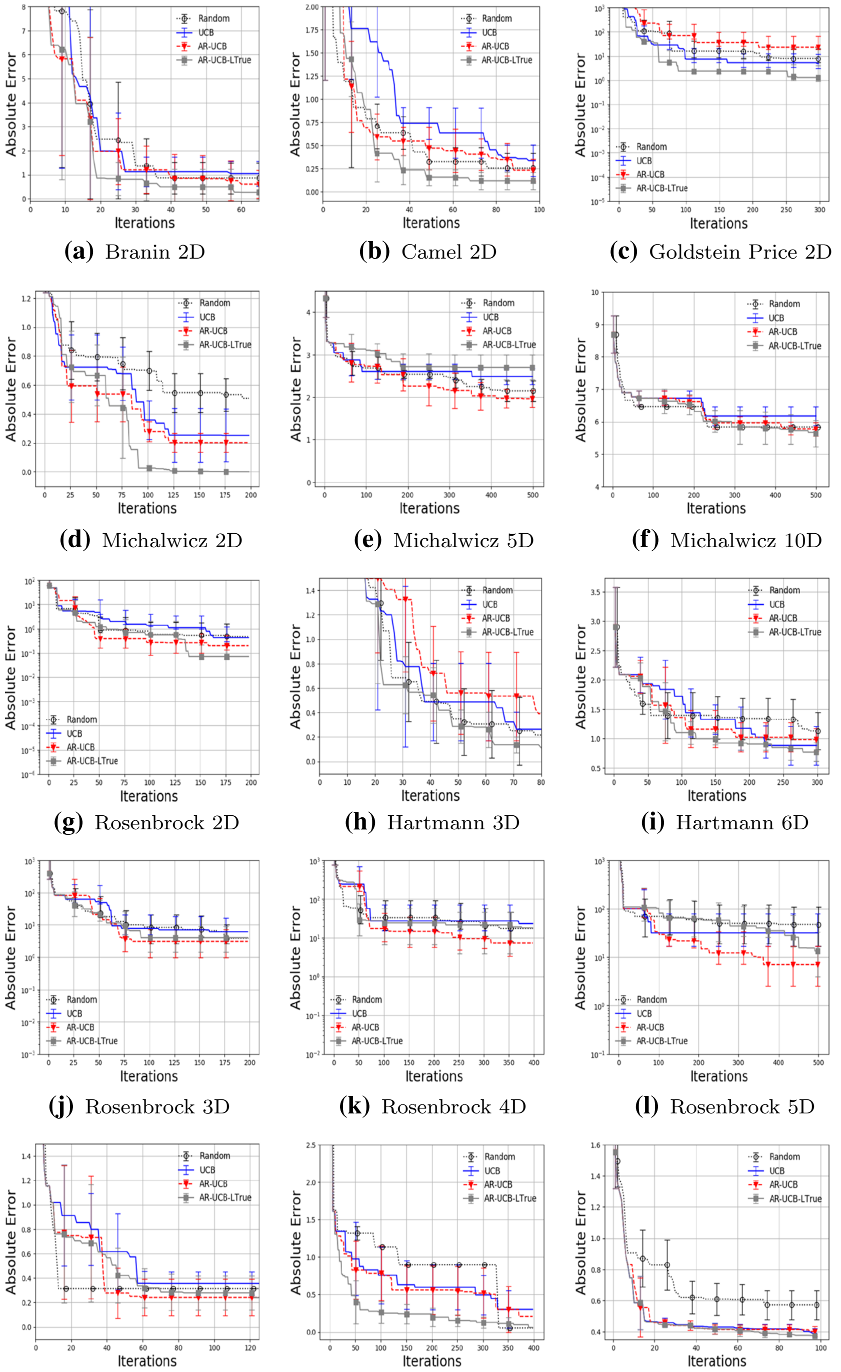
(d) Michalwicz 2D **(e)** Michalwicz 5D **(f)** Michalwicz 10D

(g) Rosenbrock 2D **(h)** Hartmann 3D **(i)** Hartmann 6D

(j) Rosenbrock 3D **(k)** Rosenbrock 4D **(l)** Rosenbrock 5D

(m) Robot pushing 3D **(n)** Robot pushing 4D **(o)** Logistic Regression

Fig. 9 Comparing the performance across the four BO and the corresponding LBO acquisition functions against Lipschitz optimization and random exploration on all the test functions (better seen in color) (Color figure online)



(a) Branin 2D **(b)** Camel 2D **(c)** Goldstein Price 2D

(d) Michalwicz 2D **(e)** Michalwicz 5D **(f)** Michalwicz 10D

(g) Rosenbrock 2D **(h)** Hartmann 3D **(i)** Hartmann 6D

(j) Rosenbrock 3D **(k)** Rosenbrock 4D **(l)** Rosenbrock 5D

(m) Robot pushing 3D **(n)** Robot pushing 4D **(o)** Logistic Regression

Fig. 10 Comparing the performance of the conventional BO acquisition function, corresponding LBO mixed acquisition function, Lipschitz optimization and random exploration for the UCB acquisition functions when using very large $\beta = 10^{16}$

References

Ahmed, M. O., Shahriari, B., & Schmidt, M. (2016). Do we need “harmless” Bayesian optimization and “first-order” Bayesian optimization? In *NIPS workshop on Bayesian optimization*.

Bardenet, R., & Kégl, B. (2010). Surrogating the surrogate: Accelerating gaussian-process-based global optimization with a mixture cross-entropy algorithm. In *International conference on machine learning (ICML), Omnipress* (pp. 55–62).

Bertsekas, D. P. (2016). *Nonlinear programming* (3rd ed.). Cambridge: MIT.

Bull, A. D. (2011). Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research*, *12*(Oct), 2879–2904.

Bunin, G. A., & François, G. (2016). Lipschitz constants in experimental optimization. arXiv preprint arXiv:1603.07847.

Chapelle, O., & Li, L. (2011). An empirical evaluation of thompson sampling. In *Advances in neural information processing systems (NIPS)* (pp. 2249–2257).

Eric, B., Freitas, N. D., & Ghosh, A. (2008). Active preference learning with discrete choice data. In *Advances in neural information processing systems (NIPS)* (pp. 409–416).

Falkner, S., Klein, A., & Hutter, F. (2017). Combining hyperband and bayesian optimization. In *NIPS workshop on Bayesian optimization*.

Gardner, J. R., Kusner, M. J., Xu, Z. E., Weinberger, K. Q., & Cunningham, J. P. (2014). Bayesian optimization with inequality constraints. In *International conference on machine learning (ICML)* (pp. 937–945).

Gelbart, M. A., Snoek, J., & Adams, R. P. (2014). Bayesian optimization with unknown constraints. arXiv preprint arXiv:1403.5607.

Ginsbourger, D., Le Riche, R., & Carraro, L. (2010). Kriging is well-suited to parallelize optimization. In Y. Tenne & C. K. Goh (Eds.), *Computational intelligence in expensive optimization problems* (pp. 131–162). Berlin: Springer.

Golovin, D., Solnik, B., Moitra, S., Kochanski, G., Karro, J., & Sculley, D. (2017). Google vizier: A service for black-box optimization. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1487–1495). ACM.

González, J., Dai, Z., Hennig, P., & Lawrence, N. (2016). Batch Bayesian optimization via local penalization. In *International conference on artificial intelligence and statistics (AISTATS)* (pp. 648–657).

Hendrix, E. M., Boglárka, G., et al. (2010). *Introduction to nonlinear and global optimization*. Berlin: Springer.

Hennig, P., & Schuler, C. J. (2012). Entropy search for information-efficient global optimization. *Journal of Machine Learning Research*, *13*(Jun), 1809–1837.

Hernández-Lobato, J. M., Gelbart, M. A., Adams, R. P., Hoffman, M. W., & Ghahramani, Z. (2016). A general framework for constrained Bayesian optimization using information-based search. *Journal of Machine Learning Research*, *17*(1), 5549–5601.

Hernández-Lobato, J. M., Hoffman, M. W., & Ghahramani, Z. (2014). Predictive entropy search for efficient global optimization of black-box functions. In *Advances in neural information processing systems (NIPS)* (pp. 918–926).

Hoffman, M. W., & Shahriari, B. (2014). Modular mechanisms for Bayesian optimization. In *NIPS workshop on Bayesian optimization* (pp. 1–5).

Hutter, F., Hoos, H. H., & Leyton-Brown, K. (2011). Sequential model-based optimization for general algorithm configuration. In *International conference on learning and intelligent optimization* (pp. 507–523). Springer.

Jamil, M., & Yang, X. S. (2013). A literature survey of benchmark functions for global optimisation problems. *International Journal of Mathematical Modelling and Numerical Optimisation*, *4*(2), 150–194.

Jones, D. R., Perttunen, C. D., & Stuckman, B. E. (1993). Lipschitzian optimization without the lipschitz constant. *Journal of Optimization Theory and Applications*, *79*(1), 157–181.

Jones, D. R., Schonlau, M., & Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, *13*(4), 455–492.

Kaelbling, L. P., & Lozano-Pérez, T. (2017). Pre-image backchaining in belief space for mobile manipulation. In H. Christensen & O. Khatib (Eds.), *Robotics research* (pp. 383–400). Cham: Springer.

Kandasamy, K., Krishnamurthy, A., Schneider, J., & Poczos, B. (2017). Asynchronous parallel Bayesian optimisation via thompson sampling. arXiv preprint arXiv:1705.09236.

Kim, J., & Choi, S. (2019). On local optimizers of acquisition functions in bayesian optimization. arXiv preprint arXiv:1901.08350.

Kushner, H. J. (1964). A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal of Basic Engineering*, *86*(1), 97–106.

Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., & Talwalkar, A. (2016). Efficient hyperparameter optimization and infinitely many armed bandits. arXiv preprint arXiv:1603.06560.

Lizotte, D. J., Greiner, R., & Schuurmans, D. (2012). An experimental methodology for response surface optimization methods. *Journal of Global Optimization*, *53*(4), 699–736.

Mahendran, N., Wang, Z., Hamze, F., & De Freitas, N. (2012). Adaptive MCMC with bayesian optimization. In *International conference on artificial intelligence and statistics (AISTATS)* (pp. 751–760).

Malherbe, C., & Vayatis, N. (2017). Global optimization of lipschitz functions. In *International conference on machine learning (ICML), Sydney, Australia, PMLR 70* (pp. 2314–2323). http://proceedings.mlr.press/v70/malherbe17a.html.

Martinez-Cantin, R., de Freitas, N., Doucet, A., & Castellanos, J. A. (2007). Active policy learning for robot planning and exploration under uncertainty. In *Robotics: Science and systems* (Vol. 3, pp. 321–328).

Močkus, J. (1975). On Bayesian methods for seeking the extremum. In *Optimization techniques IFIP technical conference* (pp. 400–404). Springer.

Pintér, J. D. (1996). *Global optimization in action* (Vol. 6)., Continuous and Lipschitz optimization: Algorithms, implementations and applications Dordrecht: Springer.

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

Qin, C., Klabjan, D., & Russo, D. (2017). Improving the expected improvement algorithm. In *Advances in neural information processing systems (NIPS)* (pp. 5387–5397).

Rasmussen, C. E., & Williams, C. K. (2006). *Gaussian processes for machine learning*. Cambridge: MIT Press.

Rios, L. M., & Sahinidis, N. V. (2013). Derivative-free optimization: A review of algorithms and comparison of software implementations. *Journal of Global Optimization*, *56*(3), 1247–1293.

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

Shahriari, B., Wang, Z., Hoffman, M. W., Bouchard-Côté, A., & de Freitas, N. (2014). An entropy search portfolio. In *NIPS workshop on Bayesian optimization*.

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

Snoek, J., Larochelle, H., Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems (NIPS)*.

Snoek, J., Rippel, O., Swersky, K., Kiros, R., Satish, N., Sundaram, N., Patwary, M., Prabhat, M., & Adams, R. (2015). Scalable Bayesian optimization using deep neural networks. In *International conference on machine learning (ICML)* (pp. 2171–2180).

Srinivas, N., Krause, A., Kakade, S. M., & Seeger, M. (2010). Gaussian process optimization in the bandit setting: No regret and experimental design. In *International conference on machine learning (ICML)* (pp. 1015–1022).

Stein, M. L. (2012). *Interpolation of spatial data: Some theory for kriging*. Berlin: Springer.

Sui, Y., Gotovos, A., Burdick, J., & Krause, A. (2015). Safe exploration for optimization with gaussian processes. In *International conference on machine learning (ICML)* (pp. 997–1005).

Thompson, W. R. (1933). On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, *25*(3/4), 285–294.

Villemonteix, J., Vazquez, E., & Walter, E. (2009). An informational approach to the global optimization of expensive-to-evaluate functions. *Journal of Global Optimization*, *44*(4), 509.

Wang, J., Clark, S. C., Liu, E., & Frazier, P. I. (2016). Parallel Bayesian global optimization of expensive functions. arXiv preprint arXiv:1602.05149.

Wang, Z., & Jegelka, S. (2017). Max-value entropy search for efficient Bayesian optimization. In *International conference on machine learning (ICML)*.

Wilson, J., Hutter, F., & Deisenroth, M. (2018). Maximizing acquisition functions for Bayesian optimization. In *NIPS* (pp. 9884–9895).

Wu, J., Poloczek, M., Wilson, A. G., & Frazier, P. (2017). Bayesian optimization with gradients. In *Advances in neural information processing systems (NIPS)* (pp. 5267–5278).