Adaptive and Safe Bayesian Optimization in High Dimensions via One-Dimensional Subspaces

Johannes Kirschner [1] **Mojmír Mutný** [1] **Nicole Hiller** [2] **Rasmus Ischebeck** [2] **Andreas Krause** [1]

Abstract

Bayesian optimization is known to be difficult to scale to high dimensions, because the acquisition step requires solving a non-convex optimization problem in the same search space. In order to scale the method and keep its benefits, we propose an algorithm (LINEBO) that restricts the problem to a sequence of iteratively chosen one-dimensional sub-problems that can be solved efficiently. We show that our algorithm converges globally and obtains a fast local rate when the function is strongly convex. Further, if the objective has an invariant subspace, our method automatically adapts to the effective dimension without changing the algorithm. When combined with the SAFEOPT algorithm to solve the sub-problems, we obtain the first safe Bayesian optimization algorithm with theoretical guarantees applicable in high-dimensional settings. We evaluate our method on multiple synthetic benchmarks, where we obtain competitive performance. Further, we deploy our algorithm to optimize the beam intensity of the Swiss Free Electron Laser with up to 40 parameters while satisfying safe operation constraints.

1. Introduction

Zero-order stochastic optimization problems arise in many applications such as hyper-parameter tuning of machine learning models, reinforcement-learning and industrial processes. An example that motivates the present work is parameter tuning of a free electron laser (FEL). FELs are large-scale physical machines that accelerate electrons in order to generate bright and shortly pulsed X-ray lasing. The X-ray pulses then facilitate many experiments in biology, medicine and material science. The accelerator and the electron beam line of a free electron laser consist of multiple individual components, each of which has several parameters that experts adjust to maximize the pulse energy. Because of different operational modes and parameter drift, this is a recurrent, time-consuming task which takes away valuable time for experiments. As a single measurement can be obtained in less than one second, the task is well suited for automated optimization with a continuous search space of about 10-100 parameters. Further, some parameters are known to physically over-parametrize the objective function, which leads to invariant subspaces and also local optima. Additionally, some settings can cause electron losses, which are required to stay below a pre-defined threshold.

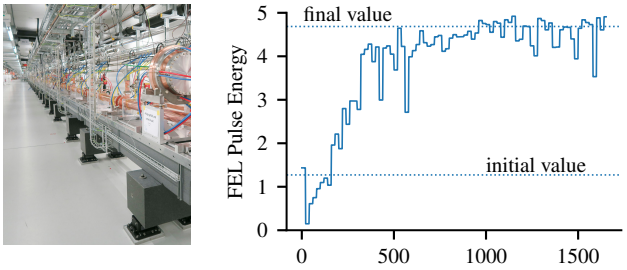


Figure 1. Left: Inside a free electron laser tunnel. Right: Using LINEBO to tune the SwissFEL pulse energy with 40 parameters.

This scenario can be cast as a gradient-free stochastic optimization problem with implicit constraints. The fact that the constraints are safety critical rules out many commonly used algorithms. Arguably, the simplest approach is to use a local optimization method with a conservatively chosen step size and a term that penalizes constraint violations in the objective, but such a method might get stuck in local optima. As an alternative, Bayesian optimization offers a principled, global optimization routine that can also operate under safety constraints (Sui et al., 2015). When applied to a low-dimensional subset of parameters, Bayesian optimization has been successfully used on FELs and in similar applications. However, it is well known that standard Bayesian optimization is difficult to scale to high-dimensional settings, because optimizing the acquisition function becomes itself an intractable optimization problem.

[1] Department of Computer Science, ETH Zurich, Switzerland [2] Paul Scherrer Institut, Switzerland. Correspondence to: Johannes Kirschner <jkirschner@inf.ethz.ch>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

In this work, we propose a novel way of using Bayesian optimization that is computationally feasible *even in high dimensions*. The key idea is to iteratively solve sub-problems of the global problem, each of which can be solved efficiently, both computationally and statistically. As feasible sub-problems we choose one-dimensional subspaces of the domain that contain the best point so far. On a one-dimensional domain, Bayesian optimization can be implemented computationally efficiently and the sample-complexity to obtain an ϵ-optimal point is independent of the outer dimension. A global GP model can nevertheless be used and allows to *share information* between the sub-problem to increase data-efficiency, in particular as samples start to accumulate close to an optimum. As we will show, our approach obtains *both local and global convergence guarantees* and further adaptively scales with the *effective dimension*, if the objective contains an invariant subspace. In the constraint setting, we use SAFEOPT to solve the sub-problems. This way, we obtain the first principled and safe Bayesian optimization algorithm applicable to high-dimensional domains.

1.1. Contributions

- We propose a novel way of using Bayesian optimization that circumvents the issue of acquisition function optimization by decomposing the global problem into a sequence of one-dimensional sub-problems that can be solved efficiently.
- Theoretically, we show that if the one-dimensional subspaces are chosen randomly, the algorithm converges with *a fast local rate* where the function is strongly convex, and converges *globally at a Lipschitz rate* that adaptively scales with the effective dimension.
- To respect safety constraints during optimization, each sub-problem can be solved with SAFEOPT. To the best of our knowledge, this is the *first principled algorithm for high dimensional safe Bayesian optimization*.
- Our algorithm is practical and amenable to heuristics that improve local convergence. As user feedback we provide one-dimensional slice plots that allow to monitor the progress and the model fit.
- We evaluate our method on synthetic benchmark functions, and apply it to tune the Swiss Free Electron Laser (SwissFEL) with up to 40 parameters on a continuous domain, satisfying safe operation constraints.

1.2. Related Work

Derivative-free stochastic optimization covers an array of algorithms from the very general grid-based methods (Nesterov, 2004; Jones, 2001) to local methods, where most of the work is spent on approximating the gradient (Nesterov & Spokoiny, 2017). Especially of interest are algorithms that optimize functions with a noisy oracle, also known as stochastic bandit feedback (Flaxman et al., 2005; Shamir, 2013). Popular examples include CMA-ES (Hansen & Ostermeier, 2001; Hansen et al., 2003), Nelder-Mead (Powell, 1973) and SPSA (Bhatnagar et al., 2013). Line-search techniques are related to our method, but have been primarily studied in the context of convex optimization (Gratton et al., 2015), also with stochastic models and search directions (Cartis & Scheinberg, 2018; Paquette & Scheinberg, 2018; Diniz-Ehrhardt et al., 2008).

Bayesian optimization is a family of algorithms using probabilistic models to determine which point to evaluate next (Mockus, 1982; Shahriari et al., 2016). Many variants appeared in literature; including GP-UCB (Srinivas et al., 2010), Thompson Sampling (Chowdhury & Gopalan, 2017), and Expected Improvement (Mockus, 1982); and recently with information theoretic criteria such as MVES or IDS (Wang & Jegelka, 2017; Kirschner & Krause, 2018). Lower bounds are known as well (Scarlett et al., 2017). Bayesian optimization on a one dimensional domain is not necessarily thought of as line search, although it can be used as such (Mahsereci & Hennig, 2017), and the one dimensional setting is theoretically well understood (Scarlett, 2018). Success stories, where Bayesian optimization outperforms classical techniques, include applications in laser technology (Schneider et al., 2018), performance optimization of Free Electron Lasers (McIntire et al., 2016a;b) and parameter optimization in the CPLEX suite (Shahriari et al., 2016).

The scaling of Bayesian optimization to high dimensions has been considered recently, as many of the commonly used kernels suffer from the curse of dimensionality. Hence, to make the problem tractable, most approaches make structural assumptions on the function such as additivity (Rolland et al., 2018; Mutný & Krause, 2018) or a low-dimensional active subspace (Djolonga et al., 2013). The latter category includes REMBO (Wang et al., 2016), which also optimizes on a random low-dimensional subspace, however, in contrast to our method, the dimension of the low-dimensional embedding needs to be known a priori. An iterative procedure to define the subspaces is proposed by Qian et al. (2016) and similarly our method relates to the Dropout-BO algorithm of Li et al. (2017a), but in both cases the convergence analysis is incomplete. A heuristic that combines local optimization with Bayesian optimization was proposed by McLeod et al. (2018).

The main instance of safe Bayesian optimization is the SAFEOPT algorithm (Sui et al., 2015; Berkenkamp et al., 2016a; Sui et al., 2018); but its formulation relies on a discretized domain, which prevents high-dimensional applications. An adaptive discretization based on particle swarms was proposed by Berkenkamp et al. (2016b).

2. Problem statement

Let $\mathcal{X} \subset \mathbb{R}^d$ be a compact domain and $f : \mathcal{X} \to \mathbb{R}$ the objective function we seek to minimize[1],

$$\min_{x \in \mathcal{X}} f(x) \quad \text{s.t.} \quad g(x) \leq 0, \tag{1}$$

where we allow for implicit constraints $g : \mathcal{X} \to \mathbb{R}$. The constraint function can be chosen vector valued in the case of multiple constraints. We refer to such constraints as *safety constraints* if it is required that the iterates x_t satisfy $g(x_t) \leq 0$ during optimization. We assume that f and g can only be accessed via a noisy oracle, that given a point $x \in \mathcal{X}$ returns an evaluation $y = f(x)+\epsilon$ and $s = g(x)+\epsilon'$, where ϵ is a noise term with sub-Gaussian tails.

Denote $f^* = \min_{x \in \mathcal{X}} f(x)$ and let $x^* \in \mathcal{X}$ be a point such that $f(x^*) = f^*$. An optimization algorithm iteratively picks a sequence of evaluations $x_1, \dots, x_T$, and obtains the corresponding noisy observations $y_1, \dots, y_T$. As a measure of progress we use *simple regret*. At any stopping time T, the optimization algorithm proposes a candidate solution $\hat{x}_T$. This point is allowed to differ from the point x_T that is chosen for the purpose of optimization, still, some algorithms might set $\hat{x}_T = x_T$. Simple regret is defined as

$$r_T := f(\hat{x}_T) - f^*, \tag{2}$$

and therefore measures the ability of an optimization algorithm to predict a minimizer at time T. To impose some regularity on f, we make the following assumption.

Assumption 1 (RKHS)**.** *The objective and constraint functions f and g are members of reproducing kernel Hilbert spaces $\mathcal{H}(k_1)$, $\mathcal{H}(k_2)$ with known kernel functions $k_1, k_2 : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and bounded norm $\|f\|_{\mathcal{H}_1}, \|g\|_{\mathcal{H}_2} \leq B$.*

This assumption is central for Bayesian optimization, as it justifies the use of Gaussian processes to estimate f from the samples (Rasmussen, 2004; Kanagawa et al., 2018).

3. Line Bayesian Optimization

In its standard formulation, Bayesian optimization uses a Gaussian process prior $\mathrm{GP}(\mu, k)$ with mean $\mu : \mathcal{X} \to \mathbb{R}$ and kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and Bayes' rule to update the posterior as observations (x_t, y_t) arrive. If a Gaussian likelihood $\epsilon \sim \mathcal{N}(0, \sigma^2)$ is used, the posterior mean $\hat{f}_t$ can be computed analytically and is equivalent to the regularized least squares kernel estimator,

$$\hat{f}_t(x) := \operatorname*{arg\,min}_{f \in \mathcal{H}_k} \sum_{t=1}^{T} \big(f(x_t) - y_t\big)^2 + \|f\|^2_{\mathcal{H}_k} .$$

[1] Bayesian optimization is typically formulated as maximization problem, but since we also have results in the flavor of convex optimization, w.l.o.g. we use minimization here.

Algorithm 1 Line Bayesian Optimization (LINEBO)

Require: Direction oracle Π, accuracy ϵ, starting point $\hat{x}_0$, Model $\mathcal{M}_0 = (\text{GP prior for } f, g)$

1: **for** $i = 1, 2, \dots, K$ **do**
2: $\quad l_i \leftarrow \Pi(\mathcal{M}_{i-1})$ *// define direction*
3: $\quad \mathcal{L}_i \leftarrow \mathcal{L}(\hat{x}_{i-1}, l_i)$ *// define subspace*
4: $\quad \hat{x}_i, \mathcal{M}_i \leftarrow \text{BayesianOptimization}(\mathcal{M}_{i-1}, \mathcal{L}_i, \epsilon)$ *// includes posterior updates (Appendix A)*
5: **end for**

From the Bayesian posterior, one can obtain credible intervals $\hat{f}_t(x) \pm \beta_t \sigma_t(x)$, which in this case are known to match frequentist confidence intervals up to the scaling factor β_t. Bayesian optimization is built upon using the uncertainty estimates σ_t, or more generally the posterior distribution, to determine promising query points x_t that efficiently reduce the uncertainty about the true maximizer x^*. Typically, an acquisition function $\alpha_t(x) := \alpha(x|\hat{f}_t, \sigma_t) : \mathcal{X} \to \mathbb{R}$ is defined to trade-off between exploration and exploitation on the GP posterior landscape and evaluations are chosen as $x_t \in \arg\max_{x \in \mathcal{X}} \alpha_t(x)$. Commonly used acquisition functions include UCB, Thompson Sampling, Expected Improvement and Max-Value Entropy Search.

The success of Bayesian optimization crucially relies on the ability to find a maximizer of the acquisition function α_t, which requires solving a non-convex optimization problem in the same search space $\mathcal{X}$. In most of the literature on Bayesian optimization, this is not discussed further as the computational cost of solving $\arg\max \alpha_t(x)$ is assumed to be negligible compared to obtaining a new evaluation on the oracle. *In practice, however, this step renders the method intractable in high-dimensional settings.*

In order to maintain tractability of the acquisition step in high dimensions, we propose to restrict the search space to a one-dimensional[2] affine subspace $\mathcal{L}(x, l) := \{x + \alpha l : \alpha \in \mathbb{R}\} \cap \mathcal{X}$, where $x \in \mathcal{X}$ is the offset, and $l \in \mathbb{R}^d$ is the direction. On such a restriction, the acquisition step can be effectively solved using an (adaptive) grid-search over $\mathcal{L}$. We will show that by carefully choosing a sequence $\mathcal{L}_1, \dots, \mathcal{L}_K$ of one-dimensional subspaces, we obtain a method that still converges globally and additionally has properties similar to a gradient method. By using a global GP model, we can share information between the sub-solvers and handle noise in a principled way.

The LINEBO method is presented in Algorithm 1. As standard for Bayesian optimization, we initialize with a GP prior. We also assume that the user provides a *direction oracle* Π, which is used to iteratively define subspaces $\mathcal{L}_i = \mathcal{L}(x_i, l_i)$. The affine subspace is always chosen to contain the previous best point to ensure a monotonic improvement over K itera-

[2] Generalization to higher dimensional subspaces is possible.

tions. We then proceed by efficiently solving the subspace $\mathcal{L}_i$ using standard Bayesian optimization (Appendix A).

A canonical example of the direction oracle is to pick the direction uniformly at random, which is also the main focus of our analysis. As we will see, this algorithm obtains both a local and a global convergence rate. Another possibility is to use (random) coordinate aligned directions, which resembles a coordinate descent algorithm. In this case, our method is a special case of DropoutUCB of Li et al. (2017a), but the global rate they obtained has a non-vanishing gap in the limit and local convergence was not analysed.

3.1. Safe Line Bayesian Optimization

The restriction of the search space allows us to effectively use a safe Bayesian optimization algorithm like SAFEOPT (see Appendix A.2) as a sub-solver, which in turn renders the global method safe (SAFELINEBO). We note that in its current formulation, SAFEOPT crucially relies on a discretized domain, which makes it difficult to apply even with $d > 3$; but it is an easy task to implement the method on a one dimensional domain. To the best of our knowledge, this way we obtain the first principled method for safe Bayesian optimization in high dimensions.

4. Convergence Analysis

4.1. Sample Complexity of 1D Bayesian Optimization

To understand the sample complexity of solving the one dimensional sub-problems, we rely on the standard analysis of Bayesian optimization developed by Srinivas et al. (2010); Abbasi-Yadkori (2012); Chowdhury & Gopalan (2017). The results are often stated in terms of a complexity measure called *maximum information gain* γ_T, which is defined as the mutual information $\gamma_T := \max_{A \subset \mathcal{X}: |A|=T} I(y_A, f_A)$. This quantity depends on the kernel and upper bounds are known for the RBF and Matern kernel (Seeger et al., 2008; Srinivas et al., 2010). We focus on a subset of kernels, which when restricted on the one dimensional affine subspace $\mathcal{L}$, their $\gamma_T(k|_{\mathcal{L}})$ satisfies the following assumption.

Assumption 2 (Bounded γ_T)**.** *Let $k : \mathbb{R} \times \mathbb{R} \to \mathbb{R}^+$ be a one-dimensional kernel and $\kappa \in (0, 0.5)$, then*

$$\gamma_T(k) \leq \mathcal{O}(T^{\kappa} \log T).$$

This is satisfied for the squared exponential kernel ($\kappa = 0$) and the Matern kernel with $\nu > \frac{3}{2}$ ($\kappa = \frac{2}{2\nu+2}$). Simple regret can be bounded as $r_T \leq \mathcal{O}(\gamma_T/\sqrt{T})$, and with the assumption above, the bound becomes $r_T \leq \mathcal{O}(T^{\kappa-1/2})$ up to logarithmic factors (see also Appendix A.1). Equivalently, the time until ϵ regret is guaranteed is $T \leq \mathcal{O}(\epsilon^{-\frac{2}{1-2\kappa}})$. The best known lower bound for this case is $r_T \geq \Omega(\epsilon^{-\frac{2}{1-\kappa}})$ (Scarlett et al., 2017), hence almost closes the gap. The overall number of evaluations after K iterations of Algorithm 1 is at most $\mathcal{O}(K\epsilon^{-\frac{2}{1-2\kappa}})$.

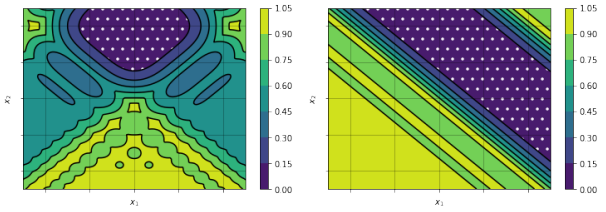


Figure 2. Function with $d_e = 2$ and $d_e = 1$. The volume of the set $V_\epsilon = \{x | f(x) - f(x^*) \leq \epsilon\}$ (dotted region) for $d_e = 2$ and $d_e = 1$ can be significantly larger if the function contains an invariant subspace, which facilitates random exploration.

4.2. Global Convergence and Subspace Adaptation

In practice, we often encounter functions that are high-dimensional but contain an (unknown) invariant subspace. This means that there are directions in which the function is constant and after removal of these dimensions the problem might not be high dimensional (see Figure 2). The dimension of the linear space where the function varies is called *effective dimension*, as formalized in the following definition.

Definition 1 (Effective dimension)**.** *The effective dimensionality of a function $f : \mathbb{R}^d \to \mathbb{R}$ is the smallest $d_e \leq d$ s.t. there exists a linear subspace $\mathcal{Y} \subset \mathbb{R}^d$ of dimension d_e and for all $x_\top \in \mathcal{Y}$ and $x_\perp \in \mathcal{Y}_\perp$, where $\mathcal{Y}_\perp$ is the orthogonal complement of $\mathcal{Y}$, $f(x_\top \oplus x_\perp) = f(x_\top \oplus 0)$.*

If Algorithm 1 is used with randomly chosen directions, we show that the convergence of the algorithm adaptively scales with the effective dimension d_e. The result is quantified in the following proposition.

Proposition 1 (Global convergence)**.** *Let f satisfy Assumption 1 with effective dimension d_e, k be twice differentiable, and let $\delta \in (0, 1)$. Then after K iterations of Algorithm 1 with accuracy ϵ and directions chosen uniformly at random, with probability at least $1 - \delta$, it holds that*

$$f(\hat{x}_K) - f^* \leq \mathcal{O}\left(\left(\frac{1}{K}\log\left(\frac{1}{\delta}\right)\right)^{\frac{2}{d_e - 1}} + \epsilon\right).$$

The proof is deferred to Appendix B.1. The result should be understood as a property of random exploration and is the best one can hope for on worst-case examples. Instances, where random search is competitive have been reported in literature (Bergstra & Bengio, 2012; Wang et al., 2016; Li et al., 2017b) and this has been attributed to the same effect. However, random search fails to control the error induced by the noise, and our method has the advantage of using the GP model to deal with the noise in a principled way.

In contrast to other algorithms that exploit subspace structure, including the REMBO algorithm of Wang et al. (2016) and SI-BO of Djolonga et al. (2013), our formulation does *not require* the knowledge of d_e in advance. Intuitively, we can demonstrate the consequence of the effective dimension and the random line algorithm by plotting the set $V_\epsilon := \{x | f(x) - f^* \leq \epsilon\}$ that appears as an isolated spike in the domain in the worst-case. For functions with an invariant subspace, the volume of the set V_ϵ increases substantially and hence the probability of a random line passing through this region increases (see Figure 2).

Naturally, such a bound cannot avoid an exponential scaling with d_e, as also does not full-scale Bayesian optimization even when restricted to the effective subspace. However, we show in the next section, that if our algorithm finds a point in the proximity of a local optimum, the convergence is dominated by a *fast* local rate, a property not exhibited by random search.

4.3. Local Convergence

By Taylor's theorem, differentiable functions have an open set around their minimizers where the function is convex or even strongly-convex. We show that if our algorithm starts in a subset of the domain where the function is strongly convex, it converges to the (local) minimum at a linear rate. Again, we focus on the instance where directions are picked at random. The key insight is that random directions can be used as descent directions in the following sense.

Lemma 1 (Random Descent Direction)**.** *Let $l \in \mathbb{R}^d$ be a uniformly random point on the d-dimensional unit sphere or uniformly among an orthonormal basis. Then,*

$$\text{for all } x \in \mathcal{X}, \qquad \mathbb{E}[\langle \nabla f(x), l \rangle^2] = \frac{1}{d} \|\nabla f(x)\|^2 .$$

The standard proof technique for descent algorithms on strongly convex functions (Nesterov, 2012) yields the following result; see Appendix B.2 for a proof.

Proposition 2. *Let f satisfy Assumption 1, be α-strongly convex and β-smooth if restricted to $\mathcal{X}_c \subset \mathcal{X}$. Let $f_c^* = \max_{x \in \mathcal{X}_c} f(x)$ and assume all iterates $\hat{x}_k$ are contained in $\mathcal{X}_c$. Then, after K iterations of Algorithm 1 with accuracy ϵ and random directions that satisfy Lemma 1, it holds that,*

$$\mathbb{E}[f(\hat{x}_K)] - f_c^* \leq \frac{\epsilon \beta d}{\alpha} + \left(1 - \frac{\alpha}{\beta d}\right)^K (f(x_0) - f_c^*).$$

To interpret the result, we fix the total number of evaluations T and assume $f_c^* = f^*$. If the kernel k restricted to any one dimensional subspace satisfies Assumption 2, we can set the accuracy $\epsilon = \left(\frac{d \log T}{2T}\right)^{(1-2\kappa)/2}$. Then, with the previous proposition, the simple regret is bounded by

$$\mathbb{E}[r_T] \leq \mathcal{O}\left(d^{3/2-\kappa} \left(\log T / T\right)^{1/2-\kappa}\right).$$

Importantly, the bound has only a *polynomial* dependence on d, for instance with the squared exponential kernel ($\kappa = 0$) we get $r_T \leq \mathcal{O}(d^{3/2}\sqrt{\log T / T})$.

4.4. Convergence under safety constraints

The ability to use an arbitrary line solver for the subproblems allows us to implement safety by using a safe BO algorithm such as SAFEOPT as a sub-solver. We call LINEBO with SAFEOPT as sub-solver SAFELINEBO. Formally, we define the safe set $\mathcal{S} = \{x \in \mathcal{X} | g(x) \leq 0\}$. It is unavoidable that an initial safe point $x_0 \in \mathcal{S}$ must be provided. The best one can hope for is the exploration of the reachable safe set $\mathcal{S}_0$, which can be defined as the connected component of $\mathcal{S}$ that contains x_0. For details, we refer to Sui et al. (2015) and Berkenkamp et al. (2016a) for multiple constraints.

The one dimensional subproblems are guaranteed to be solved safely by the guarantees of SAFEOPT under the same additional technical assumptions as for the original algorithm. However a natural question arises as to what extend the safe set is explored sufficiently when restricting the acquisition to one-dimensional subspaces. To allow for the possibility that a safe maximizer can be reached within one iteration from a given safe starting point, the straight line segment from this point to the optimum needs to be contained in $\mathcal{S}$. Naturally, this is guaranteed if the safe set $\mathcal{S}$ is convex; but other conditions are possible. For instance, if the level set $\mathcal{X}_1 = \{x : f(x) > f(x_0)\} \subset \mathcal{S}$ is both safe and convex, one can expect that the iterates do not leave $\mathcal{X}_1$ and consequently the optimum is found. Note that this is a natural condition that arises if the function is convex on a subset of domain, as we assume for our local convergence guarantees. On the other hand, it is easy to construct counterexamples even in two dimensions that are successfully solved by SAFEOPT but not with the LINEBO method (for instance with a U-shaped safe set). In practice, however, this might not be a severe limitation, in particular if constraint violations are not expected close to the optimum.

5. Practical Considerations

Our main goal is to provide a practical Bayesian optimization algorithm, with the main benefit that the acquisition step can be solved efficiently. We note that this enables the use of acquisition functions such as Thompson sampling or Max Value Entropy Search that rely on sampling the GP posterior and where an analytical expression is not available. Besides this, our methods has several further practical advantages, as we explain below.

Direction Oracles Picking random directions is one possibility to define the sub-problems, that allows us to simultaneously obtain global and local guarantees. In practice, random directions can increase variance and by in-

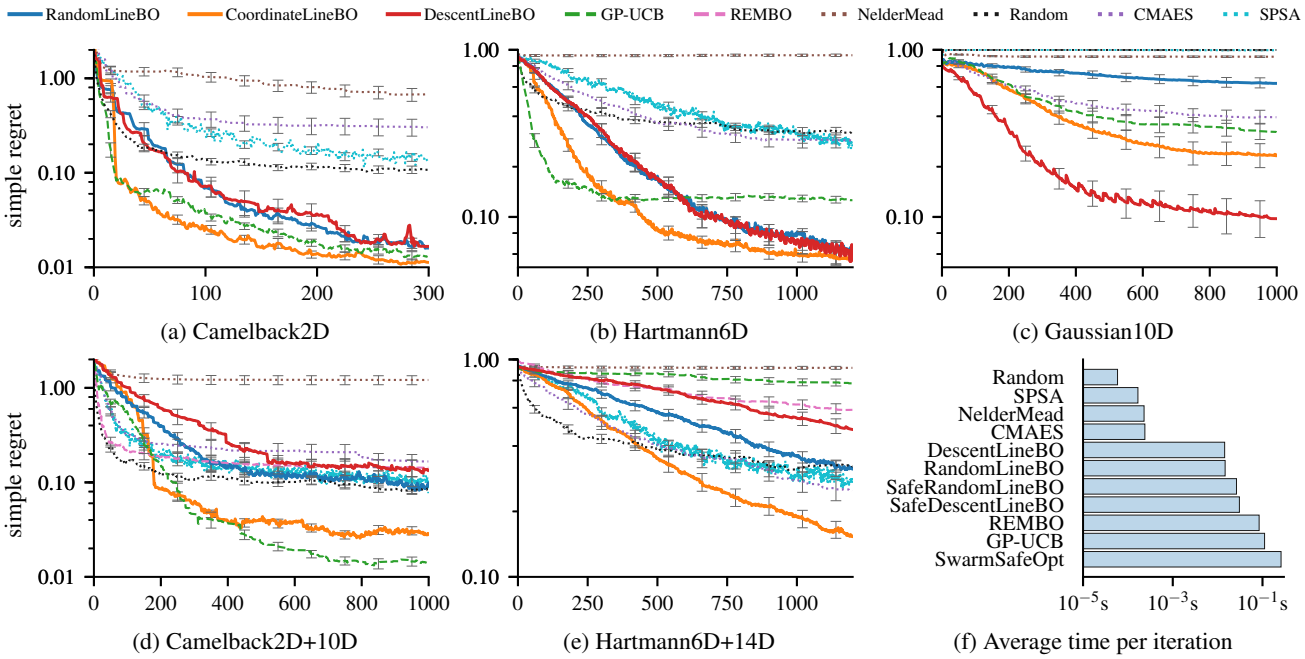


Figure 3. We compare on standard functions, Camelback (a) and Hartmann6 (b). A 10d Gaussian (c) is used to demonstrate local convergence, with a starting point such that picking up the gradient signal is difficult. We further add invariant subspaces (d, e). Figure (f) shows per-iteration computation time on a 10d benchmark. Naturally, the model-free approaches are quite fast, whereas the Bayesian optimization method have the computational burden of the GP model. However, restricting the possible acquisition space improves the per-step computation time by one order of magnitude in our implementation compared to the standard GP-UCB in our implementation.

stead choosing an (approximate) descent direction it is possible to trade-off global for local exploration. An alternative way is to choose the directions coordinate aligned (CoordinateLineBO). This we found to be efficient on many benchmark problems, likely because of reduced variance and symmetries in the objective. If one seeks to speed up local convergence, using a gradient estimate is the obvious choice. As the gradient-norm becomes smaller, one can eventually switch to random directions to encourage random exploration. For estimating descent directions, we implement the following heuristic based on Thompson Sampling. First, we take the gradient $\tilde{g}$ at $\hat{x}_i$ of a sample from the posterior GP. Then we evaluate $\hat{x}_i + \alpha\tilde{g}$, where α is a small step size, and update the model. After several such steps ($\sim d$ times), we use the gradient of the posterior mean at $\hat{x}_i$ as direction oracle (see Appendix C.2 for details). In our experiments, we found that this method (DescentLineBO) improves local convergence, and this variant was used on the free electron laser as well.

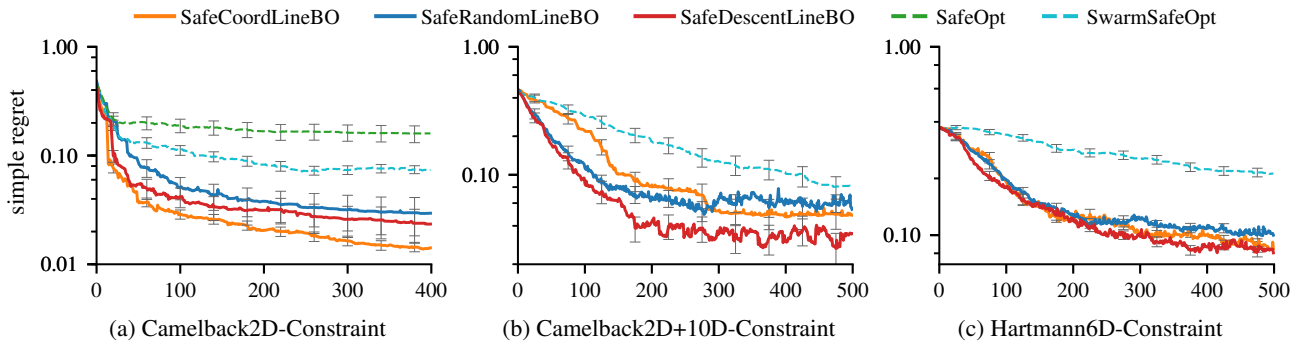
Global Model We introduced the LineBO method with a global GP model as usually done for Bayesian optimization. This has the advantage that data is *shared* between the subproblems, which can speed up convergence, but comes at the cost of inverting the kernel matrix. The iterative update cost is quadratic in the number of data points, which becomes a limiting factor typically around a few thousand steps. It is also possible to use independent sub-solvers or keep a fixed-sized data buffer; as long as the sub-problems are solved sufficiently accurately, this does not affect our theoretical guarantees and yields a further speedup.

User Feedback An additional benefit of restricting the acquisition function to a one-dimensional subspace is that we can plot evaluations together with the model predictions on this subspace. One example that we obtained when we tuned the SwissFEL is shown in Figure 5c. This allows to better understand the structure of the optimization problem; moreover, it provides valuable user-feedback, as it allows to monitor model fit and to adjust GP-hyperparameters. With safety constraints this is of particular importance, as a misspecified GP model cannot capture the safe set correctly and might cause constraint violations.

6. Empirical Evaluation

6.1. Synthetic Benchmarks

As for standard benchmarks we use the Camelback (2d) and the Hartmann6 (6d) functions. Further, we use the Gaussian $f(x) = -\exp(-4\|x\|_2^2)$ in 10 dimensions as a benchmark where local convergence is sufficient; note that when restricted to a small enough Euclidean ball this function is



(a) Camelback2D-Constraint (b) Camelback2D+10D-Constraint (c) Hartmann6D-Constraint

Figure 4. We compare on standard benchmarks with additional constraints. Note that SAFEOPT relies on a discretized domain and is therefore not applicable in the high-dimensional benchmarks. We found that performance of the methods strongly depends on the initial point, here we show average performance over a starting point chosen uniformly random in the safe set.

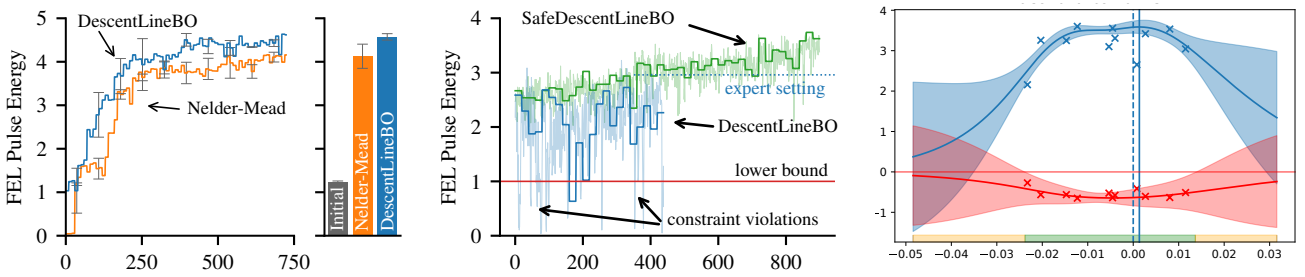
strongly convex. To obtain benchmarks with invariant subspaces, we augment the Camelback and Hartmann6 function with 10 and 14 auxiliary dimensions respectively, and shuffle the coordinates randomly. For the constraint case, we add an upper bound to the objective, ie $g(x) = -f(x) + \tau$ for some threshold τ. We found that the performance of the local methods (including our approach) depends on the initial point. For that reason, we randomized the initial points for the Camelback and Hartmann6 function uniformly in the domain; in the constrained case restricted to the safe-set. On the Gaussian function we randomize on the level set $\{x : f(x) = y_0\}$ with $y_0 = -0.2$ in the unconstrained and $y_0 = -0.4$ in the constrained case. On all experiments we add Gaussian noise with standard deviation 0.2, to obtain a similar signal-noise ratio as on our real-world application.

We compare our approach to random search, Nelder-Mead, SPSA, CMAES and standard GP-UCB. For the subspace problems we additionally compare to REMBO and its interleaved variant. The latter never perform better in our experiments and is omitted from the plots for visual clarity. In the constrained case we compare to SAFEOPT in 2 dimensions, and to the SWARMSAFEOPT heuristic on the higher-dimensional benchmarks. We use public libraries where available, details can be found in Appendix C. For our LINEBO methods, we use the UCB acquisition function. We manually chose reasonable values for hyperparameters of the methods or use recommended setting where available, but we did not run an exhaustive hyperparameter search (which would arguable not be possible in most real-world applications). All methods that use GPs share the same hyperparameters, expect on the Gaussian, where a smaller lengthscale for GP-UCB resulted in better performance.

We evaluate progress using simple regret. All regret plots show a fair comparison in terms of the total number of function evaluations on the x-axis. To compute the simple regret, each method suggests a candidate solution in each iteration (in addition to the optimization step), which is evaluated but not used in the optimization. Naturally for the GP-methods, this was chosen as the best mean of the model, and for our line methods, the best mean was determined on the current subspace. The Nelder-Mead and SPSA implementation we used did not have such an option, so progress on each evaluation is shown. Each experiment was repeated 100 times and confidence bars show the standard error.

The results for the unconstrained case are presented in Figure 3. In the standard Camelback and Hartmann6 benchmarks, we obtain competitive performance. In particular the COORDINATELINEBO method works well, which might be due to symmetries in the benchmarks. The benchmark on the Gaussian function is challenging in that the initial signal is of the same magnitude as the noise. If an optimization algorithm initially takes steps away from the optimum, the objective quickly gets very flat, and it becomes difficult to recover by means of a gradient signal only. We found that our method allow to robustly take steps towards the optimum, where local-convergence can be guaranteed, outperforming the standard GP-UCB and CMAES algorithm. Note that the DESCENTLINEBO method works particularly well on this example, as it is designed to use the estimated gradient as line directions; but it does not necessarily perform better on the other benchmarks. When adding an invariant subspace (Figure 3 d, e), our methods remain competitive with the bulk of methods, but surprisingly also UCB works very well on the camelback function with augmented coordinates. This might be due to an effect similar as in Proposition 1 carrying over from the random restarts of the approximate acquisition function optimizer.

Figure 3f shows computation time per iteration in a 10 dimensional setting (Hartmann6d+4d) averaged over 500 steps. Our methods obtain roughly one order of magnitude speed up compared to the full-scale Bayesian optimization methods; however this is of course dependent on the implementation. For GP-UCB and REMBO, we optimize the acquisition function using L-BFGS with 50 restarts, where



(a) 24 parameters (b) 24 parameters with constraint (c) User feedback

Figure 5. Experiments on the Swiss Free Electron Laser (SwissFEL). (a) Comparison of Nelder-Mead and DESCENTLINEBO. (b) Optimization with safety constraints (here DESCENTLINEBO was stopped early). (c) Slice plot provided as user feedback; these allow to monitor the GP fit and adjust hyper-parameters (red: safety constraints, blue: objective, crosses: line evaluations). Note that the model predictions are a slice of the global model, which also depends on observations from previous lines.

starting points are either randomly chosen or from a previous maximizer.

The results for the constrained case can be found in Figure 4. Our methods clearly outperform both SAFEOPT and SWARMSAFEOPT in terms of simple regret.

6.2. Tuning the Swiss Free Electron Laser

Parameter tuning is a tedious and repetitive task for operation of free electron lasers. The main objective is to increase the laser energy measured by a gas detector at the end of the beam-line. Among hundreds of available parameters that expert operators usually adjust, some parameter groups allow for automated tuning. Those include quadrupole currents settings, beam position parameters and configuration variables of the undulators. For our tests, a suitable subset of 5-40 parameters was selected by machine experts. The machine is operated at 25 Hz and we averaged 10 consecutive evaluations to reduce noise. Ideally, the computation time per step is well below 1s to avoid slowing down the overall optimization. This effectively rules out full-scale Bayesian optimization given the number of parameters. Besides manual tuning by operators, a random walk optimizer is in use and reported to often achieve satisfactory performance when run over a longer period of time; in other cases it did not improve the signal while other methods did. This hints that hill-climbing on the objective should be taken into account as a feasible step towards an acceptable solution, but global exploration and noise robustness are important, too. Nelder-Mead is mostly considered as standard benchmark in the accelerator community. Standard Bayesian optimization was previously reported to outperform it (McIntire et al., 2016a), but safety constraints, and the efficient scaling to high dimensions were not considered. Safe operation constraints include electron loss monitors and a lower threshold on the pulse energy, which is important to maintain during user operation. For our experiments we were mainly concerned with the latter, as at the time of testing, the loss monitoring system could not be used for technical reasons; but this will be an important addition once implemented.

Our results are shown in Figure 5a. To obtain a systematic comparison, we manually detuned the machine, then run both Nelder-Mead and DESCENTLINEBO twice from the same starting point (limited machine development time did not allow for a more extensive comparison). Our method soundly outperforms Nelder-Mead, both in terms of convergence speed and pulse energy at the final solution. A direct comparison between the LINEBO and SAFELINEBO in Figure 5b shows that the safe method is able to maintain the safety constraint. The safety constraint has the additional benefit of restricting the search space which we found to improve convergence in this case. The solution obtained after 600 steps (after $\sim$ 15 min) already achieves a higher pulse energy than the previous expert setting, which was obtained with the help of a local random walk optimizer. A single, successful run with 40 parameters can be found in Figure 1.

7. Conclusion

We presented a novel and practical Bayesian optimization algorithm, LINEBO, which iteratively decomposes the problem to a sequence of one dimensional sub-problems. This addresses the often ignored issue of how to maximize the acquisition function, and allows to scale the method to high-dimensional settings. We showed that the algorithm is theoretically as well as practically effective. In addition, it can also be used with safety constraints by means of safely solving each sub-problem, and is therefore, to the best of our knowledge, the first method to achieve this. Finally, we demonstrated how we apply the SAFELINEBO method on SwissFEL for tuning the pulse energy with up to 40 parameters on a continuous domain while satisfying safe operation constraints.

Acknowledgements

The authors thank in particular Manuel Nonnenmacher for the work he did during his Master thesis and Kfir Levy for valuable discussions and feedback. For the experiments on the free electron laser, the authors would like to acknowledge the support of the entire SwissFEL team.

This research was supported by SNSF grant 200020_159557 and 407540_167212 through the NRP 75 Big Data program. Further, this project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme grant agreement No 815943.

References

Abbasi-Yadkori, Y. *Online Learning for Linearly Parametrized Control Problems*. PhD thesis, 2012.

Bergstra, J. and Bengio, Y. Random search for hyper-parameter optimization. *JMLR*, 2012.

Berkenkamp, F., Krause, A., and Schoellig, A. P. Bayesian optimization with safety constraints: Safe and automatic parameter tuning in robotics. Technical report, arXiv, February 2016a.

Berkenkamp, F., Moriconi, R., Schoellig, A. P., and Krause, A. Safe learning of regions of attraction for uncertain, nonlinear systems with Gaussian processes. In *Proc. of the IEEE Conference on Decision and Control*, pp. 4661–4666, 2016b.

Bhatnagar, S., Prasad, H., and Prashanth, L. *Stochastic Recursive Algorithms for Optimization*. Springer London, London, 2013. ISBN 978-1-4471-4285-0. doi: 10.1007/978-1-4471-4285-0_1.

Cartis, C. and Scheinberg, K. Global convergence rate analysis of unconstrained optimization methods based on probabilistic models. *Mathematical Programming*, 169 (2):337–375, Jun 2018. ISSN 1436-4646. doi: 10.1007/s10107-017-1137-4.

Chowdhury, S. R. and Gopalan, A. On kernelized multi-armed bandits. In *International Conference on Machine Learning (ICML)*, 2017.

Diniz-Ehrhardt, M., Martínez, J., and Raydán, M. A derivative-free nonmonotone line-search technique for unconstrained optimization. *Journal of computational and applied mathematics*, 219(2):383–397, 2008.

Djolonga, J., Krause, A., and Cevher, V. High-dimensional Gaussian process bandits. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 1025–1033, 2013.

Flaxman, A. D., Kalai, A. T., and McMahan, H. B. Online convex optimization in the bandit setting: gradient descent without a gradient. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 385–394, 2005.

GPy. GPy: A gaussian process framework in python. http://github.com/SheffieldML/GPy, 2012.

Gratton, S., Royer, C. W., Vicente, L. N., and Zhang, Z. Direct search based on probabilistic descent. *SIAM Journal on Optimization*, 25(3):1515–1541, 2015.

Hansen, N. and Ostermeier, A. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001. doi: 10.1162/106365601750190398.

Hansen, N., Müller, S. D., and Koumoutsakos, P. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). *Evolutionary Computation*, 11(1):1–18, March 2003. ISSN 1063-6560. doi: 10.1162/106365603321828970.

Jones, D. R. Direct global optimization algorithmdirect global optimization algorithm. In *Encyclopedia of optimization*, pp. 431–440. Springer, 2001.

Kanagawa, M., Hennig, P., Sejdinovic, D., and Sriperumbudur, B. K. Gaussian processes and kernel methods: A review on connections and equivalences. *Technical report, arXiv:1807.02582*, 2018.

Kirschner, J. and Krause, A. Information directed sampling and bandits with heteroscedastic noise. In *Proc. International Conference on Learning Theory (COLT)*, July 2018.

Li, C., Gupta, S., Rana, S., Nguyen, V., Venkatesh, S., and Shilton, A. High dimensional bayesian optimization using dropout. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pp. 2096–2102, 2017a.

Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., and Talwalkar, A. Hyperband: A novel bandit-based approach to hyperparameter optimization. *The Journal of Machine Learning Research*, 18(1):6765–6816, 2017b.

Mahsereci, M. and Hennig, P. Probabilistic line searches for stochastic optimization. *JMLR*, 18:1–59, 2017.

McIntire, M., Cope, T., Ratner, D., and Ermon, S. Bayesian optimization of fel performance at lcls. *Proceedings of IPAC2016*, 2016a.

McIntire, M., Ratner, D., and Ermon, S. Sparse Gaussian processes for Bayesian optimization. In *Uncertainty in Artificial Intelligence*, 2016b.

McLeod, M., Roberts, S., and Osborne, M. A. Optimization, fast and slow: Optimally switching between local and bayesian optimization. In *International Conference on Machine Learning*, pp. 3440–3449, 2018.

Mockus, J. The bayesian approach to global optimization. *System Modeling and Optimization*, pp. 473–481, 1982.

Mutný, M. and Krause, A. Efficient high dimensional bayesian optimization with additivity and quadrature fourier features. In *Neural and Information Processing Systems (NeurIPS)*, December 2018.

Nesterov, Y. Introduction to convex optimization: A basic course. *Springer*, 2004.

Nesterov, Y. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

Nesterov, Y. and Spokoiny, V. Random gradient-free minimization of convex functions. *Foundations of Computational Mathematics*, 17(2):527–566, Apr 2017. ISSN 1615-3383. doi: 10.1007/s10208-015-9296-2.

Paquette, C. and Scheinberg, K. A stochastic line search method with convergence rate analysis. *arxiv*, 2018.

Powell, M. J. D. On search directions for minimization algorithms. *Mathematical Programming*, 4(1):193–201, Dec 1973. ISSN 1436-4646. doi: 10.1007/BF01584660.

Qian, H., Hu, Y.-Q., and Yu, Y. Derivative-free optimization of high-dimensional non-convex functions by sequential random embeddings. In *International Joint Conferences on Artificial Intelligence (IJCAI)*, 2016.

Rasmussen, C. E. Gaussian processes in machine learning. In *Advanced lectures on machine learning*, pp. 63–71. Springer, 2004.

Rolland, P., Scarlett, J., Bogunovic, I., and Cevher, V. High-dimensional bayesian optimization via additive models with overlapping groups. In *International Conference on Artificial Intelligence and Statistics*, pp. 298–307, 2018.

Scarlett, J. Tight regret bounds for Bayesian optimization in one dimension. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pp. 4500–4508. PMLR, 10–15 Jul 2018.

Scarlett, J., Bogunovic, I., and Cevher, V. Lower bounds on regret for noisy gaussian process bandit optimization. In *Conference on Learning Theory*, pp. 1723–1742, 2017.

Schneider, P.-I., Garcia Santiago, X., Soltwisch, V., Hammerschmidt, M., Burger, S., and Rockstuhl, C. Benchmarking five global optimization approaches for nano-optical shape optimization and parameter reconstruction. *arXiv preprint arXiv:1809.06674*, 2018.

Seeger, M. W., Kakade, S. M., and Foster, D. P. Information consistency of nonparametric gaussian process methods. *IEEE Transactions on Information Theory*, 54(5):2376–2382, 2008.

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

Shamir, O. On the complexity of bandit and derivative-free stochastic convex optimization. In *Conference on Learning Theory*, pp. 3–24, 2013.

Srinivas, N., Krause, A., Kakade, S. M., and Seeger, M. Gaussian process optimization in the bandit setting: No regret and experimental design. *International Conference on Machine Learning*, 2010.

Steinwart, I. and Christmann, A. *Support vector machines*. Springer Science & Business Media, 2008.

Sui, Y., Gotovos, A., Burdick, J., and Krause, A. Safe exploration for optimization with gaussian processes. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pp. 997–1005. PMLR, 2015.

Sui, Y., Burdick, J., Yue, Y., et al. Stagewise safe bayesian optimization with gaussian processes. In *International Conference on Machine Learning*, pp. 4788–4796, 2018.

Wang, Z. and Jegelka, S. Max-value entropy search for efficient Bayesian optimization. *International Conference on Machine Learning*, 2017.

Wang, Z., Hutter, F., Zoghi, M., Matheson, D., and de Feitas, N. Bayesian optimization in a billion dimensions via random embeddings. *Journal of Artificial Intelligence Research*, 55:361–387, 2016.

A. Bayesian Optimization

A general outline of Bayesian optimization is given in Algorithm 2 below. The evaluation point is determined using an acquisition function $\alpha(x|\mathcal{M})$ that typically depends on the GP model $\mathcal{M}$. One of the most common acquisition functions, that has been analyzed theoretically, is GP-UCB (Srinivas et al., 2010). With posterior mean $\hat{f}_t$ and posterior standard deviation $\sigma_t(x)$ it is defined as the lower confidence bound (in the minimization setting),

$$\alpha(x) = \hat{f}_t(x) - \beta_t \sigma_t(x) \tag{3}$$

for a scaling factor β_t (for details, see Srinivas et al. (2010)).

Algorithm 2 Bayesian Optimization (BO)

Require: Domain $\mathcal{X}$, accuracy ϵ, acquisition function α
GP prior $\mathcal{M}_0 = GP(\mu_0, k_0)$
for $t = 1, 2, 3, \dots$ **do**
  $x_t \leftarrow \arg\min_{x \in \mathcal{X}} \alpha(x|\mathcal{M}_{t-1})$ *// acquisition step*
  $y_t \leftarrow f(x_t) + \epsilon$ *// obtain observation*
  $\mathcal{M}_t \leftarrow \mathcal{M}_{t-1}|(x_t, y_t)$ *// update posterior*
  $\hat{x}_t = \arg\min_{x \in \mathcal{X}} err(x)$ *// best point, eq. 4*
  if $err(\hat{x}_t) \leq \epsilon$ **then**
    return $\hat{x}_t, \mathcal{M}_t$ *// best point, posterior model*
  end if
end for

A.1. Sample Complexity of Bayesian Optimization

In the frequentist analysis with confidence intervals $\hat{f}(x) \pm \sigma(x)$, the simple regret at any point $x \in \mathcal{X}$ can be controlled with

$$err(x) := \hat{f}(x) + \sigma(x) - (\min_{x' \in \mathcal{X}} \hat{f}(x') - \sigma(x')) \,. \tag{4}$$

The sample complexity bounds of GP-UCB make sure that the breaking condition in Algorithm 2 is eventually satisfied. Even though these bounds are typically formulated for cumulative regret, the proofs in fact bound the following quantity,

$$\sum_{t=1}^{T} err(x_t) \leq \mathcal{O}(\gamma_T \sqrt{T}) \,,$$

see (Chowdhury & Gopalan, 2017, Theorem 3). From this a bound on simple regret follows as

$$r_T \leq err(\hat{x}_T) \leq \frac{1}{T} \sum_{t=1}^{T} err(x_t) \leq \frac{\gamma_T}{\sqrt{T}} \,.$$

Bounds on γ_T are known for different kernels, including for the linear kernel: $\gamma_T \leq \mathcal{O}(d\sqrt{T})$, the RBF kernel: $\gamma_T \leq \mathcal{O}((\log(T))^{d+1})$, and the Matern kernel with $\nu > 1$: $\gamma_T \leq \mathcal{O}\big(T^{d(d+1)/(2\nu + d(d+1))}(\log(T))\big)$; see Srinivas et al. (2010, Theorem 5).

A.2. Safe Bayesian Optimization

SAFEOPT uses an idea that is similar to Algorithm 2. A GP-model is used to estimate the implicit constraint function g with confidence intervals $\hat{g}(x) \pm \beta_t \sigma_t^g(x)$. The confidence estimates can be used to define a conservatively estimated safe set $\hat{\mathcal{S}} = \{x \in \mathcal{X} : \hat{g}_t(x) + \beta_t \sigma_t^g(x) \leq 0\}$. The acquisition step is then restricted to $\hat{\mathcal{S}}$, and under the condition that the confidence estimates hold, the algorithm does not violate the constraints. However, the exploration problem becomes more difficult, as both f and g need to be explored in an appropriate way. For completeness, we reproduce the pseudo-code from (Sui et al., 2015; Berkenkamp et al., 2016a) in Algorithm 3. Please refer to the original publication for a more detailed treatment.

In each iteration t, the algorithm defines a safe set $\mathcal{S}_t$, the set of potential minimizers M_t, and the expander set G_t with points that can possibly enlarge the safe set. The algorithm uses two functions; the first is the uncertainty at a specific point $x \in \mathcal{X}$ to determine which points to acquire,

$$w_t(x) = \max(2\beta_t \sigma_t^f(x), 2\beta_t \sigma_t^g(x)) \,. \tag{5}$$

Next, to quantify possible expanders of the safe set,

$$p_t(x) = |\{x \in \mathcal{X} \setminus \mathcal{S}_t | \hat{g}_t(x) + \beta \sigma_t^g(x) - L \left\|x\right\|_2 \geq h\}| \,. \tag{6}$$

Also, denote $u_t(x) = \hat{f}_t(x) + \beta_t \sigma_t^f(x)$ and $l_t(x) = \hat{f}_t(x) - \beta_t \sigma_t^f(x)$ the lower and upper confidence bound of f.

Algorithm 3 SAFEOPT

Require: Domain $\mathcal{X}$, initial safe set $\mathcal{S}_0$, safety threshold h, Lipschitz constant L, GP priors $\mathcal{M}_0$ for both f and g,
for $t = 1, 2, 3, \dots$ **do**
  $\mathcal{S}_t \leftarrow \cup_{x \in S_{t-1}} \{x' \in \mathcal{X} | \hat{g}_t(x) - L \left\|x - x'\right\|_2 \geq h\}$ *// Enlarge safe set*
  $G_t \leftarrow \{x \in \mathcal{S}_t | p_t(x) > 0 \text{ as in (6)}\}$ *// expander set*
  $M_t \leftarrow \{x \in \mathcal{S}_t | l_t(x) \leq \min_{x' \in S_t} u_t(x')\}$ *// plausible minimizer set*
  $x_t \leftarrow \arg\max_{x \in G_t \cup M_t} w_t(x)$ *// uncertainty sampling*
  $y_t \leftarrow f(x_t) + \epsilon$ *// acquire observations*
  $s_t \leftarrow g(x_t) + \epsilon$
  $\mathcal{M}_t \leftarrow \mathcal{M}_{t-1}|(x_t, y_t, s_t)$ *// update posterior*
  $\hat{x}_t = \arg\min_{x \in \mathcal{X}} err(x)$ *// best point, eq. 4*
  if $err(\hat{x}_t) \leq \epsilon$ **then**
    return $\hat{x}_t, \mathcal{M}_t$ *// best point, posterior model*
  end if
end for

The algorithm in its original formulation requires the knowledge of a Lipschitz constant L. It is however possible to derive a bound on L from the norm bound $\|f\|_{\mathcal{H}}$ as by Assumption 1. Note that this algorithm is in particularly

simple to implement in the one-dimensional setting. There, the safe-set is always an interval with its endpoints being possible expanders.

B. Proofs of Theoretical Results

B.1. Global Convergence

We first show the following lemma.

Lemma 2. *Let $f \in \mathcal{H}_k$ be twice differentiable with effective dimension d_e. Further, let $f_K^* = \min_{x \in \mathcal{L}_1, \dots, \mathcal{L}_K} f(x)$ be the minimum objective value that can be obtained by minimizing any line up to iteration K. Then,*

$$\mathbb{P}[f_K^* - f^* \leq \tau] \geq 1 - \exp(-K\xi(\tau)) \tag{7}$$

where $\xi(\tau)$ is a lower bound on the probability that a random line intersects the set $V_\tau = \{x \in \mathcal{X} : f(x) \leq f^* + \tau\}$. Further, if the first order condition at the minimum x^* is met, then $\xi(\tau) = \Omega\left(\tau^{\frac{d_e - 1}{2}}\right)$

Before we go on to the proof, we show how Proposition 1 follows.

Proof of Proposition 1. Denote $f_K^* = \min_{x \in \mathcal{L}_1, \dots, \mathcal{L}_K} f(x)$ and remember that each line is solved up to ϵ accuracy, therefore $f(\hat{x}_K) \leq f_K^* + \epsilon$. Using Lemma 2, we know that with probability at least $1 - \exp(-K\xi(\tau))$, $f(\hat{x}) - f^* \leq \epsilon + \tau$, hence when $\exp(-K\xi(\tau)) = \delta$ the statement in the proposition is true. Solving for τ yields $\tau \leq \mathcal{O}\left(\frac{1}{K} \log\left(\frac{1}{\delta}\right)\right)^{2/(d_e - 1)}$, concluding the proof. □

Proof of Lemma 2. Let $\xi(\tau)$ be a lower bound on the probability that a random line intersects the set $V_\tau = \{x \in \mathcal{X} : f(x) \leq f^* + \tau\}$. Using this, we find

$$\begin{aligned} & \mathbb{P}[f_K^* - f^* \geq \tau] \\ = \; & \mathbb{P}[f_1^* - f^* \geq \tau \wedge \dots \wedge f_K^* - f^* \geq \tau] \\ = \; & \prod_{i=1}^{K} \mathbb{P}[f_i^* - f^* \geq \tau | x_{i-1}, y_{i-1}, \dots, x_1, y_1] \\ \leq \; & (1 - \xi(\tau))^K \leq \exp(-K\xi(\tau)) \end{aligned}$$

where the last inequality uses $1 - x \leq e^{-x}$. Hence

$$\mathbb{P}[f_K^* - f^* \leq \tau] \geq 1 - (1 - \xi(\tau))^K \geq 1 - e^{-\xi(\tau)K} .$$

Using the assumption that f is twice-differentiable, we can over-approximate f around a minimizer x^* using a quadratic function $f(x^* + h) \leq f(x^*) + \frac{\alpha}{2} \|h\|^2$ for small h and $\alpha > 0$. Therefore $\tilde{V}_\tau := \{x \in \mathcal{X} | \frac{\alpha}{2} \|x - x^*\|^2 \leq \tau\}) \subset V_\tau$, and it is enough to intersect $\tilde{V}_\tau$ with a random line. Note that if we also use the assumption that the function varies only in d_e dimensions, we can restrict the approximation to the active subspace, therefore $\mathrm{Vol}(\tilde{V}_\tau) \geq \Omega(\tau^{d_e/2})$. If we allow the hidden constant also to depend on the diameter of the domain $\mathcal{X}$, the probability that a random line intersects $\tilde{V}_\tau$, and therefore V_τ, is at least $\Omega(\tau^{(d_e - 1)/2})$. □

Remark 1. *The assumption that f is twice differentiable is always satisfied if the kernel function k is twice differentiable, see Steinwart & Christmann (2008, Corollary 4.36).*

B.2. Local Convergence

First, we recall the definition of smooth and strongly convex functions.

Definition 2 (Strong Convexity)**.** *A differentiable function $f : \mathcal{X} \to \mathbb{R}$ is called α-strongly convex if there exists $\alpha > 0$ such that for all$x, h \in \mathcal{X} \subseteq \mathbb{R}^d$,*

$$\langle \nabla f(x), h \rangle + \frac{\alpha}{2} \|h\|_2^2 \quad \leq \quad f(x + h) - f(x) \tag{8}$$

Definition 3 (Smoothness)**.** *A differentiable function $f : \mathcal{X} \to \mathbb{R}$ is called β-smooth, if there exists $\beta > 0$ such that for all $x, h \in \mathcal{X} \subseteq \mathbb{R}^d$,*

$$f(x + h) - f(x) \quad \leq \quad \langle \nabla f(x), h \rangle + \frac{\beta}{2} \|h\|_2^2 . \tag{9}$$

Strong convexity implies the Polyak–Lojasiewicz condition,

$$f(x) - f(x^*) \leq \frac{1}{2\alpha} \langle \nabla f(x), \nabla f(x) \rangle . \tag{10}$$

The next lemma shows that randomly chosen directions can be used as descent directions (Lemma 1 in the main text).

Lemma 3 (Random Descent Direction)**.** *Let $l \in \mathbb{R}^d$ be a randomly chosen direction. Specifically assume that l is uniformly random on the d-dimensional unit sphere (random directions) or uniformly among an orthonormal basis (coordinate descent). Then for any $g \in \mathbb{R}^d$*

$$\mathbb{E}[\langle g, l \rangle^2] = \frac{1}{d} \|g\|^2 .$$

Proof of Lemma 3. Denote by l_i the ith coordinate of l. Note that $\mathbb{E}[\sum l_i^2] = 1$, hence $\mathbb{E}[l_i^2] = \frac{1}{d}$. Further $\mathbb{E}[l_i l_j] = \mathbb{E}[\mathbb{E}[l_i | l_j]] = 0$ for $i \neq j$ due to symmetry argument if l is uniformly on the sphere, and by orthonormality in the coordinate case. The result follows from expanding the square and using the previous two equations. □

Lemma 4 (Exact line search oracle)**.** *Let f be α-strongly convex and β-smooth on a domain $\mathcal{X}$. If we obtain iterates $\hat{x}_i$ from Algorithm 1 with random directions l_i that satisfy Lemma 3, then the exact line-search solution $x_{i+1}^* = \arg\min_{x \in \mathcal{L}_i} f(x)$ improves per step by*

$$\mathbb{E}[f(x_{i+1}^*) - f(x^*)] \leq \left(1 - \frac{\alpha}{\beta d}\right) (f(\hat{x}_i) - f(x^*)) .$$

Proof of Lemma 4. Let $x^*_{i+1} = \arg\min_{x \in \mathcal{L}_i} f(x)$ be the solution obtained from an exact line-search on the subproblem $\mathcal{L}_i$. This implies that for any $h \in \mathbb{R}$,

$$f(x^*_{i+1}) - f(\hat{x}_i) \leq f(\hat{x}_i + h l_i) - f(\hat{x}_i)$$

Further assume that the directions l_i are random, satisfy Lemma 3 and $\|l_i\| = 1$. Smoothness implies that

$$f(\hat{x}_i + h l_i) - f(\hat{x}_i) \overset{\text{Def (3)}}{\leq} \langle \nabla f(\hat{x}_i), h l_i \rangle + \frac{\beta}{2} h^2 \|l_i\|_2^2$$

In particular, the inequality also holds for $h = -\frac{\langle \nabla f(\hat{x}_i), l_i \rangle}{\beta}$, and note that $l_i \in \mathbb{R}^d$ is normalized, ie $\|l_i\| = 1$, hence taking the previous two inequalities together,

$$f(x^*_{i+1}) - f(\hat{x}_i) \leq -\frac{\langle \nabla f(\hat{x}_i), l_i \rangle^2}{2\beta}$$

Taking expectation over the random direction $l_i \in \mathbb{R}^d$ and using Lemma 3, we get

$$\begin{aligned} \mathbb{E}[f(x^*_{i+1})] - f(\hat{x}_i) &\leq -\frac{1}{2\beta d} \|\nabla f(x_i)\|_2^2 \\ &\leq -\frac{\alpha}{\beta d} (f(\hat{x}_i) - f(x^*)) \,, \end{aligned}$$

and the last inequality uses the Polyak–Lojasiewicz condition (10). Rearranging concludes the proof,

$$\mathbb{E}[f(x^*_{i+1}) - f(x^*)] \leq \left(1 - \frac{\alpha}{\beta d}\right) (f(\hat{x}_i) - f(x^*)) \,.$$

□

Proof of Proposition 2. Assume that we run Algorithm 2 with accuracy ϵ and obtain iterates $\hat{x}_i$ that do not leave the subset $\mathcal{X}_c$ where the function is β-smooth and α-strongly convex. Denote the exact line-search solutions by $x^*_{i+1} = \arg\min_{x \in \mathcal{L}_{i+1}} f(x)$ and $\gamma = \frac{\alpha}{\beta d}$, to find

$$\begin{aligned} \mathbb{E}[f(\hat{x}_{i+1})] - f(x^*) &\leq \mathbb{E}[f(x^*_{i+1}) - f(x^*)] + \epsilon \\ &\leq (1-\gamma)(f(\hat{x}_i) - f(x^*)) + \epsilon \,, \end{aligned}$$

by means of Lemma 4. Recursively applying the previous inequality gives

$$\begin{aligned} \mathbb{E}[f(x_K)] - f(x^*) &\leq \epsilon \sum_{i=0}^{K-1} (1-\gamma)^i \\ &\quad + (1-\gamma)^K (f(\hat{x}_0) - f(x^*)) \\ &\leq \frac{\epsilon}{\gamma} + (1-\gamma)^K (f(\hat{x}_0) - f(x^*)) \\ &\leq \frac{\epsilon}{\gamma} + \exp(-K\gamma)(f(x_0) - f(x^*)) \end{aligned}$$

This concludes the proof. □

C. Synthetic Experiments

C.1. Implementation Details

For the RANDOM method, we pick points uniformly in the domain and report the best observation as candidates, without any control on the noise. UCB is implemented using GPy (GPy, 2012), and the acquisition function is maximized using the L-BFGS solver provided by the SciPy library. To evade local maxima, 50 restarts are used, both containing random points and previous maximizers. For Nelder-Mead we use the SciPy implementation. SPSA is provided by noisyopt (https://noisyopt.readthedocs.io). CMAES is provided by the pycma package (https://github.com/CMA-ES/pycma). Our REMBO and InterleavedREMBO implementation is based on https://github.com/jmetzen/bayesian_optimization. SafeOpt and SwarmSafeOpt use the author implementation https://github.com/befelix/SafeOpt.

C.2. Direction Oracle

Our DESCENTLINEBO algorithm uses the following heuristic to find the directions. We use a step-size of $\alpha = 0.1$ and $m = 2d$ evaluations in our experiments. Note that this can be seen as Thompson sampling on the Euclidean ball $B_\alpha(\hat{x})$ with a linear approximation of the posterior GP.

Algorithm 4 Descent Direction Oracle

Require: Current point $\hat{x}$, step size α, number of evaluations m, GP model $GP(\mu_0, k_0)$.

for $i = 1, 2, \ldots, m$ **do**
  $\tilde{f} \sim GP(\mu_{i-1}, k_{i-1})$
  $\tilde{g} \leftarrow \nabla \tilde{f}(\hat{x})$ *// sample gradient at x*
  $x_i \leftarrow \hat{x} - \alpha \tilde{g}$ *// linear approximation on $B_\alpha(\hat{x})$*
  $y_i \leftarrow f(x_t) + \epsilon$ *// obtain observation*
  $\mu_i, k_i \leftarrow (\mu_{i-1}, k_{i-1}) | (x_t, y_t)$ *// update posterior*
end for
return $\nabla \mu_m(\hat{x})$ *// gradient of posterior mean*