Unexpected Improvements to Expected Improvement for Bayesian Optimization

Anonymous Author(s)
Affiliation
Address
email

Abstract

Expected Improvement (EI) is arguably the most popular acquisition function in Bayesian optimization and has been applied successfully in practice, but its performance is often exceeded by that of more recent methods. However, EI and its variants, including for the parallel and multi-objective settings, are very challenging to optimize due to the fact that their acquisition values are numerically zero in many regions, resulting in inconsistencies in EI's performance across the literature. This difficulty generally increases as the number of observations, dimensionality of the search space, or the number of constraints grow. We propose a new family of acquisition functions, LogEI, whose members either have identical or approximately equal optima as their canonical counterparts, but are substantially easier to optimize numerically. We demonstrate that numerical pathologies manifest themselves in "classic" analytic EI, as well as its variants, including constrained EI, parallel EI, and expected hypervolume improvement, and propose corresponding reformulations that avoid these pathologies. Our empirical results show that members of the LogEI family of acquisition functions achieve substantially better performance than their canonical counterparts and surprisingly, are on par with or exceed the performance of recent state-of-the-art acquisition functions, highlighting the understated role of numerical optimization in the literature.

1 Introduction

Bayesian Optimization (BO) is a popular framework for sample-efficient black-box optimization of expensive-to-evaluate functions [15, 18]. BO leverages a probabilistic *surrogate model* in conjunction with an *acquisition function* to determine where to query the underlying objective function. Improvement-based acquisition functions, such as Expected Improvement (EI) and Probability Improvement (PI), are among the earliest and most widely used acquisition functions for efficient global optimization of non-convex functions [30, 42]. EI has been extended to the constrained [17, 19], noisy [38], and multi-objective [13] setting, as well as their respective batch variants [2, 8, 60], and is a standard baseline in the BO literature [15, 50]. While much of the literature has focused on developing new sophisticated acquisition functions, subtle yet critical implementation details of foundational BO methods are often overlooked. Notably, the performance of EI and its variants can vary greatly from one implementation to another, leading to inconsistent results even for *mathematically identical* formulations. To our knowledge, these inconsistencies have not been detailed in the literature, although the problem of optimizing EI effectively has been discussed in various works (e.g., [15, 21, 60]).

In this work, we identify pathologies of several improvement-based acquisition functions and propose reformulations utilizing careful numerical implementations that lead to increases in the optimization performance of improvement-based acquisition functions which often match or exceed that of recent methods. Our main contributions are:

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

1. We introduce LogEI, a new family of acquisition functions whose members either have identical or approximately equal optima as their canonical counterparts, but are substantially easier to optimize numerically. Notably, the analytic variant of LogEI, which results in the same BO policy as EI (mathematically), empirically shows significantly improved optimization performance.
2. We extend the ideas behind analytical LogEI to a number of other members of the EI family, including constrained EI (cEI), Expected Hypervolume Improvement (EHVI), as well as their respective batch variants for parallel BO, qEI and qEHVI, using smooth approximations of the acquisition utilities to obtain non-zero gradients.
3. We demonstrate that our newly proposed acquisition functions substantially outperform their respective analogues on a broad range of benchmarks without incurring meaningful additional computational cost.

While we focus on Expected Improvement, the same issues and similar solutions also apply to Probability of Improvement [35] and its variants (see Appendix A).

Motivation

Maximizing acquisition functions in BO is a challenging problem. The optimization surface is generally non-convex, and often contains many local maxima. While zeroth-order methods are sometimes used, on continuous domains gradient-based methods tend to be far more effective at optimizing acquisition functions, especially in higher dimensions.

In addition to the challenges stemming from non-convexity that are shared across acquisition functions, improvement-based acquisition functions are particularly challenging to optimize because their value and gradient can be minuscule in large swaths of its domain. Although EI is never *mathematically* zero under a Gaussian posterior distribution (except at previously evaluated points in the case of noiseless observations), it often is *exactly* zero numerically due to floating point precision. The same applies to its gradient, making EI (and PI, see Appendix A) difficult to optimize via gradient-based methods. Figure 1 illustrates this behavior on a simple one-dimensional problem.

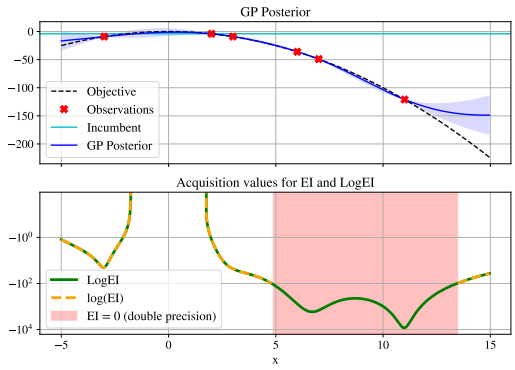

Figure 1: Values of EI and LogEI on a quadratic objective. For points at which the likelihood of improving over the incumbent is small, EI takes on extremely small values. Notably, EI is *exactly* zero numerically (in double floating point precision) in a large part of the domain ($\approx [5, 13.5]$). As Figure 2 to the right shows, this behavior gets worse as the dimension of the problem and the number of observed data points grow, rendering gradient-based optimization of EI futile.

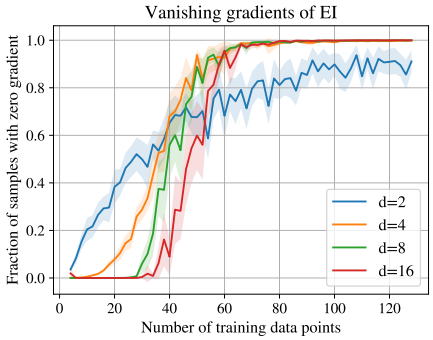

Figure 2: Empirical fraction of points randomly sampled from the domain for which the gradient of EI is approximately numerically zero ($< 10^{-10}$) as a function of the number of (randomly generated) data points n for different dimensions d on the Ackley. As the model uncertainty shrinks with more data, EI and its gradients become numerically zero across most of the domain.

To increase the chance of finding the global optimum of non-convex functions, gradient-based optimization is typically performed from multiple starting points [54]. A multi-start gradient-based approach can indeed help avoid getting stuck in local optima. However, for improvement-based acquisition functions, optimization becomes increasingly challenging as more data is collected and the likelihood of improving over the incumbent diminishes (see our theoretical results in Section 3 and the empirical illustration in Figure 2). As a result, gradient-based optimization with multiple random

starting points will eventually degenerate into random search when the gradients at the starting points are numerically zero. This problem is particularly acute in high dimensions and for objectives with a large range.

Various initialization heuristics have been proposed to address this behavior by modifying the random-restart strategy. Rather than starting from random candidates, an alternative naïve approach would be to use initial conditions close to the best previously observed inputs. However, doing that alone inherently limits the initialization heuristic to a type of local search, which cannot have global guarantees. To attain such guarantees, it is necessary to use an asymptotically space-filling heuristic; even if not random, this will entail evaluating the acquisition function in regions where no prior observation lies. Ideally, these regions should permit gradient-based optimization of the objective for efficient acquisition function optimization, which necessitates the gradients to be non-zero. In this work, we show that this can be achieved for a number of improvement-based acquisition functions, and demonstrate empirically how this leads to substantially improved BO performance.

2 Background

We consider the problem of maximizing an expensive-to-evaluate black-box function $f_{\text{true}} : \mathbb{X} \mapsto \mathbb{R}^M$ over some feasible set $\mathbb{X} \subset \mathbb{R}^d$. Suppose we have collected data $\mathcal{D}_n = \{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathbb{X}$ and $y_i = f_{\text{true}}(x_i) + v_i(x_i)$ with v_i some noise corrupting the true function value $f_{\text{true}}(x_i)$. The response f_{true} may be multi-output (e.g. in the case of multiple objectives or black-box constraints), in which case $y_i, v_i \in \mathbb{R}^M$. We use Bayesian optimization (BO), which relies on a surrogate model f that for any *batch* $\mathbf{x} := \{x_1, \dots, x_q\}$ of candidate points provides a probability distribution over the outputs $f(\mathbf{x}) := (f(x_1), \dots, f(x_q))$. The acquisition function α then utilizes this posterior prediction to assign an acquisition value to $\mathbf{x}$ that quantifies the value of evaluating the points in $\mathbf{x}$, trading off exploration and exploitation.

2.1 Gaussian Processes

Gaussian Processes (GP) models [49] are the most widely used surrogates in BO, due to their high data efficiency and good uncertainty quantification. For our purposes, it suffices to consider a GP as a mapping that provides a multivariate Normal distribution over the outputs $f(\mathbf{x})$ for any $\mathbf{x}$:

$$f(\mathbf{x}) \sim \mathcal{N}(\boldsymbol{\mu}(\mathbf{x}), \boldsymbol{\Sigma}(\mathbf{x})), \qquad \boldsymbol{\mu} : \mathbb{X}^q \to \mathbb{R}^{qM}, \quad \boldsymbol{\Sigma} : \mathbb{X}^q \to \mathcal{S}_+^{qM}. \tag{1}$$

In the single-outcome ($M = 1$) setting, $f(\mathbf{x}) \sim \mathcal{N}(\mu(\mathbf{x}), \Sigma(\mathbf{x}))$ with $\mu : \mathbb{X}^q \to \mathbb{R}^q$ and $\Sigma : \mathbb{X}^q \to \mathcal{S}_+^q$. In the sequential ($q = 1$) case, this further reduces to a univariate Normal distribution: $f(x) \sim \mathcal{N}(\mu(x), \sigma^2(x))$ with $\mu : \mathbb{X} \to \mathbb{R}$ and $\sigma : \mathbb{X} \to \mathbb{R}_+$.

2.2 Expected Improvement and (some of) its variants

Expected Improvement In the fully-sequential ($q = 1$), single-outcome ($M = 1$) setting, "classical" EI [43] is defined as

$$\text{EI}_{y^*}(x) = \mathbb{E}_{f(x)}\big[[f(x) - y^*]_+\big] = \sigma(x)\, h\left(\frac{\mu(x) - y^*}{\sigma(x)}\right), \tag{2}$$

where $[\cdot]_+$ denotes the $\max(0, \cdot)$ operation, y^* is the *incumbent* (the best function value observed so far), $h(z) = \phi(z) + z\Phi(z)$, and ϕ, Φ are the standard Normal density and distribution functions, respectively. This formulation is arguably the most widely used acquisition function in BO, and the default in many popular software packages.

Constrained Expected Improvement The *constrained BO* setting involves one or more black-box constraints; the problem is typically formulated as finding $\max_{x \in \mathbb{X}} f_{\text{true},1}(x)$ such that $f_{\text{true},i}(x) \leq 0$ for $i \in \{2, \dots, M\}$. Feasibility-weighting the improvement [17, 19] is a natural way to formulate constrained EI:

$$\text{cEI}_{y^*}(x) = \mathbb{E}_{f(x)}\Big[[f_1(x) - y^*]_+ \, \Pi_{i=2}^M \mathbf{1}\{f_i(x) \leq 0\}\Big], \tag{3}$$

where $\mathbf{1}$ is the indicator function. If the constraints are modeled as conditionally independent of the objective f_1 this can be further simplified as the product of EI and the probability of feasibility.

Parallel Expected Improvement In many settings, one may evaluate f_{true} on $q > 1$ candidates in parallel to increase throughput. The associated batch analogue of EI [20, 58] is given by

$$\text{qEI}_{y^*}(\mathbf{x}) = \mathbb{E}\Big[\max_{j=1,\dots,q}\big\{[f(x_j) - y^*]_+\big\}\Big]. \tag{4}$$

Unlike EI, qEI does not admit a closed-form expression and thus is typically computed using Monte Carlo sampling [2, 58]:

$$\text{qEI}_{y^*}(\mathbf{x}) \approx \sum_{i=1}^{N} \max_{j=1,\dots,q}\big\{[\xi_j^i(\mathbf{x}) - y^*]_+\big\}, \tag{5}$$

where $\xi^i(\mathbf{x}) \sim f(\mathbf{x})$ are random samples drawn from the (joint) model posterior at $\mathbf{x}$.

Expected Hypervolume Improvement In multi-objective optimization (MOO), there generally is no single best solution; instead the goal is to explore the Pareto Frontier between multiple competing objectives (the set of mutually-optimal objective vectors) . A common measure of the quality of a (finitely approximated) Pareto Frontier $\mathcal{P}$ between M objectives with respect to a specified reference point $\mathbf{r} \in \mathbb{R}^M$ is its *hypervolume* $\text{HV}(\mathcal{P}, \mathbf{r}) := \lambda\big(\bigcup_{i=1}^{|\mathcal{P}|}[\mathbf{r}, y_i]\big)$, where $[\mathbf{r}, y_i]$ denotes the hyper-rectangle bounded by vertices $\mathbf{r}$ and y_i, and λ is the Lebesgue measure. Thus, a natural acquisition function to optimize is the expected hypervolume improvement (EHVI)

$$\text{EHVI}(\mathbf{x}) = \mathbb{E}\big[\text{HV}(\mathcal{P} \cup f(\mathbf{x}), \mathbf{r})] - \text{HV}(\mathcal{P}, \mathbf{r})\big] \tag{6}$$

from obtaining a set $\mathcal{Y} \sim f(\mathbf{x})$ of new observations. If $q = 1$ and the objectives are modeled with independent GPs, EHVI can be expressed in closed form [63]. In the general case, Monte Carlo approximations are used (qEHVI) [8].

2.3 Optimizing acquisition functions

Optimizing an acquisition function (AF) is a challenging task that amounts to solving a non-convex optimization problem, to which multiple approaches and heuristics have been applied. These include gradient-free methods such as divided rectangles [29], evolutionary methods such as CMA-ES [22], first-order methods such as stochastic gradient ascent (see e.g., Daulton et al. [9], Wang et al. [58]), and (quasi-)second order methods [15] such as L-BFGS-B [6]. Multi-start optimization is commonly employed with gradient-based methods to mitigate the risk of getting stuck in local minima. Initial points for optimization are selected via various heuristics with different levels of complexity, ranging from simple uniform random selection to `BoTorch`'s initialization heuristic, which selects initial points by performing Boltzmann sampling on a set of random points according to their acquisition function value [2]. See Appendix B for a more complete account of initialization strategies and optimization procedures used by popular implementations. We focus on gradient-based optimization as often leveraging gradients results in faster and more performant optimization [8].

Optimizing AFs for parallel BO that quantify the value of a batch of $q > 1$ points is more challenging than optimizing their sequential counterparts due to the higher dimensionality of the optimization problem (qd instead of d) and the more challenging optimization surface. A common approach to simplify the problem is to use a *sequential greedy* strategy that greedily solves a sequence of single point selection problems. For $i = 1, \dots, q$, candidate x_i is selected by optimizing the AF (for $q = 1$) conditional on the previously selected designs $x_1, ..., x_{i-1}$ and their unknown observations (e.g. by fantasizing the values at those designs) [60]. For submodular AFs, including EI, PI, and EHVI, using a sequential greedy strategy will incur no more than $1/e$ regret compared to joint optimization, and previous works have found that sequential greedy optimization yields *improved* BO performance compared to joint optimization [8, 60]. Contrary to these findings, we show that with our reformulations joint batch optimization is indeed superior to the sequential greedy strategy.

2.4 Related Work

While there is a substantial body of work introducing a large variety of different AFs, much less focus has been on the question of how to effectively implement and optimize these AFs.

Zhan and Xing [64] provide a comprehensive review of a large number of different variants of the EI family, but do not discuss any numerical or optimization challenges. Zhao et al. [65] propose to combine a variety of different initialization strategies to select initial conditions for optimization of

acquisition functions and show empirically that this improves optimization performance. However, they do not address any potential issues or degeneracies with the acquisition functions themselves. Recent works have considered effective gradient-based approaches for acquisition optimization. Wilson et al. [60] demonstrates how stochastic first-order methods can be leveraged for optimizing Monte Carlo acquisition functions. Balandat et al. [2] build on this work and propose to use a sample average approximation to MC acquisition functions that admits gradient-based optimization using deterministic higher-order optimizers such as L-BFGS-B.

Another line of work proposes to switch from BO to local optimization based on some stopping criterion to achieve faster local convergence, using either zeroth order [44] or gradient-based [41] optimization. While McLeod et al. [41] are also concerned with numerical issues, we emphasize that those issues arise due to ill-conditioned covariance matrices and are orthogonal to the numerical pathologies of improvement-based acquisition functions.

3 Theoretical Analysis of Expected Improvement's Vanishing Gradients

In this section, we shed light on the conditions on the objective function and surrogate model that give rise to the numerically vanishing gradients in EI, as seen in Figure 2. In particular, we show that as a BO algorithm closes the optimality gap $f^* - y_n^*$ (here f^* is the global maximum of the function f_{true}) and the associated GP surrogate's uncertainty decreases, it becomes exceedingly likely for EI to exhibit numerically vanishing gradients.

Given a distribution P_x over the inputs x, the probability that the argument $(\mu(x) - y^*)/\sigma(x)$ to h in Eq. (2) is smaller than a threshold B exceeds $P_x(f(x) < f^* - \epsilon_n)$, with high probability, where ϵ_n depends on the optimality gap $f^* - y_n^*$ and the maximum posterior uncertainty $\max_x \sigma_n(x)$. This pertains to the problem of numerically vanishing values and gradients of EI, since the numerical support $\mathcal{S}_\eta(h) = \{x : |h(x)| > \eta\}$ of a naïve implementation of h in (2) is limited by a lower bound $B(\eta)$ that depends on the floating point precision η. Formally, $\mathcal{S}_\eta(h) \subset [B(\eta), \infty)$ even though $\mathcal{S}_0(h) = \mathbb{R}$ mathematically. As a consequence, the following result can be seen as a bound on the probability of encountering numerically vanishing values and gradients in EI using samples from the distribution P_x to initialize the acquisition function optimization.

Lemma 1. *Suppose f is drawn from a Gaussian process prior P_f, $y^* \leq f^*$, μ_n, σ_n are the mean and standard deviation of the posterior $P_f(f|\mathcal{D}_n)$ and $B \in \mathbb{R}$. Then with probability $1 - \delta$,*

$$P_x\left(\frac{\mu_n(x) - y_n^*}{\sigma_n(x)} < B\right) \geq P_x\left(f(x) < f^* - \epsilon_n\right) \tag{7}$$

where $\epsilon_n = (f^* - y_n^*) + \left(\sqrt{-2\log(2\delta)} - B\right) \max_x \sigma_n(x)$.

For any given – and especially early – iteration, ϵ_n does not have to be small, as both the optimality gap and the maximal posterior standard deviation can be large initially. Note that under certain technical conditions on the kernel function and the asymptotic distribution of the training data $\mathcal{D}_n$, the maximum posterior variance vanishes guaranteeably as n increases, see [36, Corollary 3.2]. On its own, Lemma 1 gives insight into the non-asymptotic behavior by exposing a dependence to the distribution of objective values f. In particular, if the set of inputs that give rise to high objective values ($\approx f^*$) is concentrated, $P(f(x) < f^* - \epsilon)$ will decay very slowly as ϵ increases, thereby maintaining a lower bound on the probability of close to 1. As an example, this is the case for the Ackley function, especially as the dimensionality increases, which explains the behavior in Figure 2.

4 Improving Expected Improvement (and its cousins)

In this section, we propose modifications to and re-formulations of analytic and MC-based improvement-based acquisition functions that render them significantly easier to optimize. In the following, we will use differing fonts, e.g. $\log$ and `log`, to differentiate between the mathematical functions and their numerical implementations.

4.1 Analytic LogEI

Implementations of "classic" analytic EI exhibit numerically zero values and gradients even though EI and its gradient are mathematically nonzero on the entire real line, except in the noiseless case

at the previously evaluated points. However, if implemented naïvely, h is numerically zero when $(\mu(x)-y^*)/\sigma(x)$ is small, which happens when the model has high confidence that little improvement can be achieved at x.

We propose an implementation of $\log \circ h$ that can be accurately computed for a much larger range of inputs than a naïve implementation $\texttt{h}$ or $\texttt{log} \circ \texttt{h}$. Specifically, we compute analytic

$$\text{LogEI}_{y^*}(x) = \texttt{log\_h}((\mu(x) - y^*)/\sigma(x)) + \texttt{log}(\sigma(x)), \tag{8}$$

where $\texttt{log\_h}$ is mathematically equivalent to $\log \circ h$ and can be stably and accurately computed by

$$\texttt{log\_h}(z) = \begin{cases} \texttt{log}(\phi(z) + z\Phi(z)) & z > -1 \\ -z^2/2 - c_1 + \texttt{log1mexp}(\texttt{logerfcx}(-z/\sqrt{2})|z| + c_2) & z < -1 \end{cases} \tag{9}$$

where $c_1 = \log(2\pi)/2$, and $c_2 = \log(\pi/2)/2$, and $\texttt{log1mexp}$, $\texttt{logerfcx}$ are numerically stable implementations of $\log(1 - \exp(x))$ and $\log(\exp(x^2)\text{erfc}(x))$, respectively (see Appendix A for details). Notably, the asymptotically quadratic behavior of $\texttt{log\_h}$ becomes apparent in the second case, making the function particularly amenable to gradient-based optimization. This has *significant* practical implications for BO using EI, as evidenced by the empirical results in Section 5. Numerically vanishing values and gradients affect – as far as we are aware – all public implementations of EI.

4.2 Monte Carlo Parallel LogEI

For Monte Carlo formulations of Parallel EI that perform differentiation on the level of MC samples, the situation is worse in that they exhibit not just numerically, but mathematically zero gradients for a significant proportion of practically relevant inputs. For qEI, the primary issue is taking the discrete maximum over the q outcomes for each MC sample in (5). In particular, the acquisition utility of expected improvement in Eq. 4 on a single sample ξ_i of f is $\max_j [\xi_i(x_j) - y^*]_+$. Mathematically, we smoothly approximate the acquisition utility in two stages: 1) $u_{ij} = \text{softplus}_{\tau_0}(\xi_i(x_j) - y^*) \approx [\xi_i(x_j) - y^*]_+$ and 2) $\|u_{i\cdot}\|_{1/\tau_{\max}} \approx \max_j u_{ij}$. Since the resulting quantities are strictly positive, they can be transformed to log-space permitting an implementation of qLogEI that is numerically stable and can be optimized effectively, similar to the analytic case. In particular,

$$\begin{aligned} \text{qLogEI}_{y^*}(\mathbf{x}) &= \log \int \left(\textstyle\sum_j \text{softplus}_{\tau_0}(f(x_j) - y^*)^{1/\tau_{\max}} \right)^{\tau_{\max}} df \\ &\approx \texttt{logsumexp}_i(\tau_{\max} \texttt{logsumexp}_j(\texttt{logsoftplus}_{\tau_0}(\xi^i(x_j) - y^*))/\tau_{\max})), \end{aligned} \tag{10}$$

where i is the index of the Monte Carlo draws from the GP posterior, $j = 1, \ldots, q$ is the index for the candidate in the batch, and $\texttt{logsoftplus}$ is a numerically stable implementation of $\log(\log(1 + \exp(x)))$. See Appendix A for additional details.

While the smoothing in (10) approximates the original qEI formulation, the following result shows that the resulting approximation error can be quantified and tightly bounded as a function of the temperature parameters τ_0, $\tau_{\max}$ and the batch size q. See Appendix C for the proof.

Lemma 2. *[Approximation Guarantee] Given the temperature parameters τ_0 and $\tau_{\max}$, the approximation error of* qLogEI *to* qEI *is bounded by*

$$\left|\exp(\text{qLogEI}(x)) - \text{qEI}(x)\right| \leq \log(2)\ \tau_0 + q^{\tau_{\max}} - 1. \tag{11}$$

In Appendix D, we show the importance of setting the temperatures sufficiently low for qLogEI to achieve good optimization characteristics, something that only becomes possible by transforming all involved computations to log-space. Otherwise, the smoothed approximation to the acquisition utility (e.g., using a regular $\texttt{softplus}$ function) would similarly exhibit numerically vanishing gradients, as is the case mathematically for the discrete $\max$ operator.

4.3 Constrained EI

Both analytic and Monte Carlo variants of LogEI can be extended for optimization problems with black-box constraints. For analytic cEI with independent constraints of the form $f_i(x) \leq 0$, the constrained formulation in Eq. (3) simplifies to $\text{logcEI}(x) = \text{LogEI}(x) + \sum_i \log(P(f_i(x) \leq 0))$, which can be readily and stably computed using LogEI in Eq. (8) and, if f_i is modelled by a GP, a stable implementation of the Gaussian log cumulative distribution function. For the Monte Carlo variant, we apply a similar strategy as for Eq. (10) to the constraint indicators in Eq. (3): 1) a smooth approximation via a sigmoid and 2) an accurate and stable implementation of its log value (see Appendix A for details).

4.4 Monte Carlo Parallel LogEHVI

The numerical difficulties of qEHVI in (6) are similar to those of qEI, and the basic ingredients of smoothing and log-transformations still apply, but the details are significantly more complex since qEHVI uses many operations that have mathematically zero gradients with respect to some of the inputs. Our implementation is based on the differentiable inclusion-exclusion formulation of the hypervolume improvement [8]. As a by-product, the implementation also readily allows for the differentiable computation of the expected log hypervolume, instead of the log expected hypervolume, note the order, which can be preferable in certain applications of multi-objective optimization [16].

5 Empirical Results

We compare standard versions of analytic EI (EI) and constrained EI (cEI), Monte Carlo parallel EI (qEI), as well as Monte Carlo EHVI (qEHVI). We also compare with other state-of-the-art baselines, namely lower-bound Max-Value Entropy Search (GIBBON) [45] and single- and multi-objective Joint Entropy Search (JES) [25, 55]. All experiments are implemented using BoTorch [2] and utilize multi-start optimization of the AF with `scipy`'s L-BFGS-B optimizer. In order to avoid conflating the effect of BoTorch's default initialization strategy with those of our contributions, we use 16 initial points chosen uniformly at random from which to start the L-BFGS-B optimization (for a comparison with other initialization strategies, see the Appendix D). We run multiple replicates and report mean and error bars of ± 2 standard errors of the mean. Additional details can be found in Appendix D.1.

Single-objective, sequential BO We compare EI and LogEI on the 10-dimensional convex Sum-of-Squares (SoS) function $f(x) = \sum_{i=1}^{10} (x_i - 0.5)^2$, using 20 restarts seeded from 1024 pseudo-random samples through BoTorch's default initialization heuristic. Figure 3 shows that due to vanishing gradients, EI is unable to make progress even on this trivial problem.

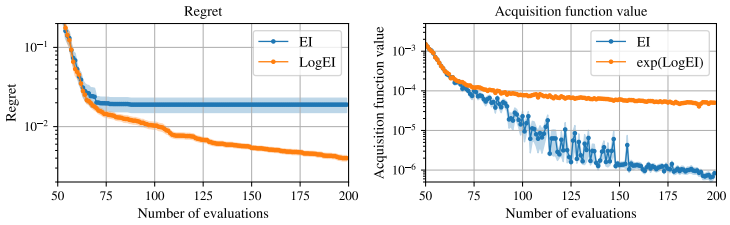

Figure 3: Regret and EI acquisition value for the candidates selected by maximizing EI and LogEI on the convex Sum-of-Squares problem. Optimization stalls out for EI after about 75 observations due to vanishing gradients (indicated by the jagged behavior of the acquisition value), while LogEI continues to make steady progress.

In Figure 4, we compare performance on the Ackley and Michalewicz test functions [51]. Notably, LogEI substantially outperforms EI on Ackley as the dimensionality increases. Ackley is a challenging multimodal function for which it is critical to trade off local exploitation with global exploration, a task made exceedingly difficult by the numerically vanishing gradients of EI in a large fraction of the search space. We see a similar albeit less pronounced behavior on Michalewicz, which reflects the fact that Michalewicz is a somewhat less challenging problem than Ackley.

BO with Black Box Constraints Figure 5 shows results on four engineering design problems with black box constraints, where LogcEI massively outperforms the naive cEI implementation. Similar to the unconstrained problems, the performance gains of LogcEI over cEI grow with increasing dimensionality of the problems, as well as with the number of constraints. Notably, while running these benchmarks we found that for some problems, LogcEI in fact improved upon some of the best results quoted in the original literature that used up to almost three orders of magnitude more function evaluations (see Appendix D).

Parallel Expected Improvement with qLogEI Figure 6 reports optimization performance of parallel BO on the 16-dimensional Ackley and Levy functions for both sequential greedy and joint

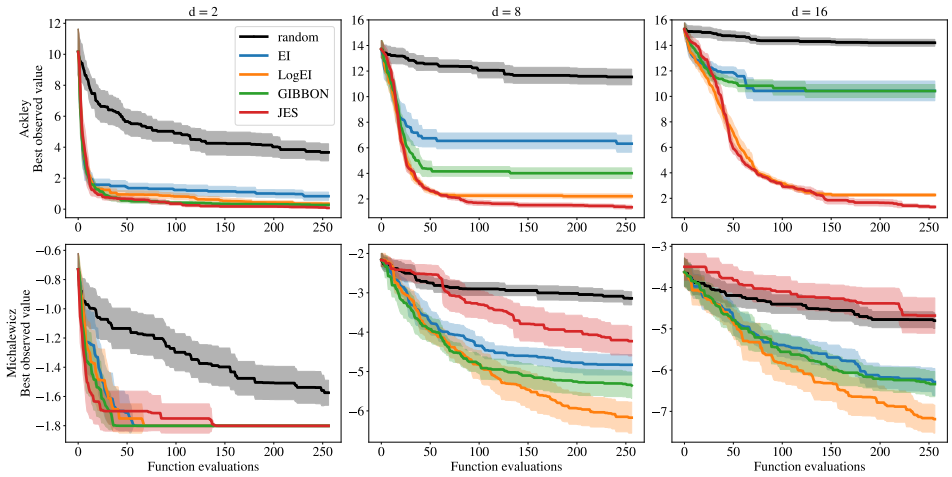

Figure 4: Best objective value as a function of iterations on the moderately and severely non-convex Michalewicz and Ackley problems for varying numbers of input dimensions. LogEI substantially outperforms both EI and GIBBON, and this gap widens as the problem dimensionality increases. JES performs slightly better than LogEI on Ackley, but for some reason fails on Michalewicz. Notably, JES is almost two orders of magnitude slower than the other acquisition functions (see Appendix D).

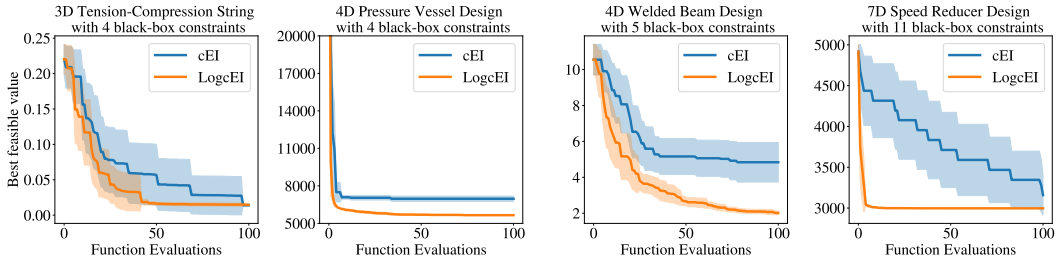

Figure 5: Best feasible objective value as a function of number of function evaluations (iterations) on four engineering design problems with black-box constraints after an initial $2d$ pseudo-random evaluations.

batch optimization. Besides the apparent substantial advantages of qLogEI over qEI, they key observation here is that, in contrast to previous findings in the literature [60], jointly optimizing the candidates of batch acquisition functions can yield highly competitive optimization performance.

Multi-Objective optimization with qLogEHVI Figure 7 compares qLogEHVI and qEHVI on 6 different test problems with 2 or 3 objectives, and ranging from 2-30 dimensions. This includes 3 real world inspired problems: cell network design for optimizing coverage and capacity [12], laser plasma acceleration optimization [26], and vehicle design optimization [40, 52]. The results are consistent with our findings in the single-objective and constrained cases: qLogEHVI consistently outperforms qEHVI, and the gap is larger on higher dimensional problems. See the Appendix D for problem details.

6 Conclusion

Our results demonstrate that the problem of vanishing gradients is a major source of the difficulty of optimizing improvement-based acquisition functions and that we can mitigate this issue through careful reformulations and implementations. As a result, we see substantially improved optimization performance across a variety of modified EI variants across a broad range of problems. In particular, contrary to previous findings, we demonstrate that joint batch optimization for parallel BO outperforms the sequential greedy approach typically used in practice (which also benefits from our modifications). Besides the convincing performance gains, one of the key advantages of our

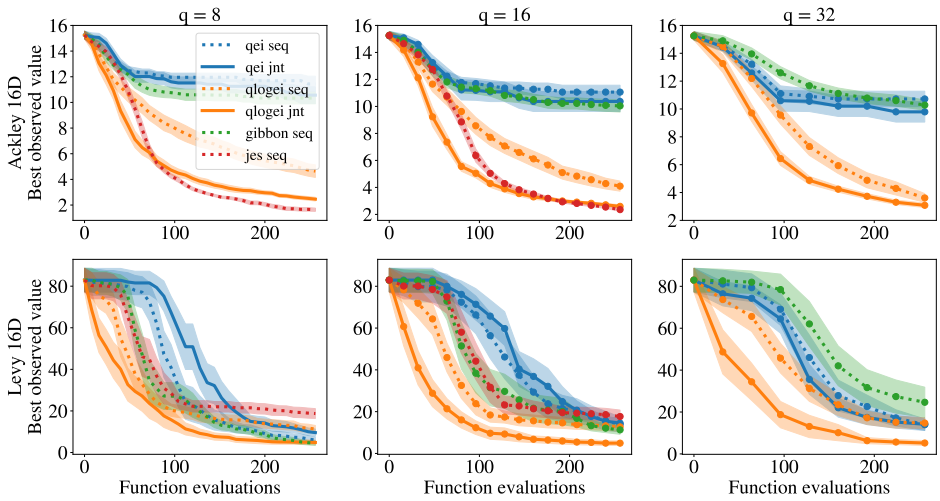

Figure 6: Best objective value for parallel BO as a function of the number evaluations for single-objective optimization in 16 dimensions with varying batch sizes q. The advantage of qLogEI over qEI grows as the dimensionality increases and, in accordance with Lem. 1, the differences are less pronounced on the Levy function for which the distribution of inputs with good objective values is less concentrated in the search space. Notably and surprisingly, joint optimization of the batch outperforms sequential greedy optimization, overturning previous findings for canonical qEI [60].

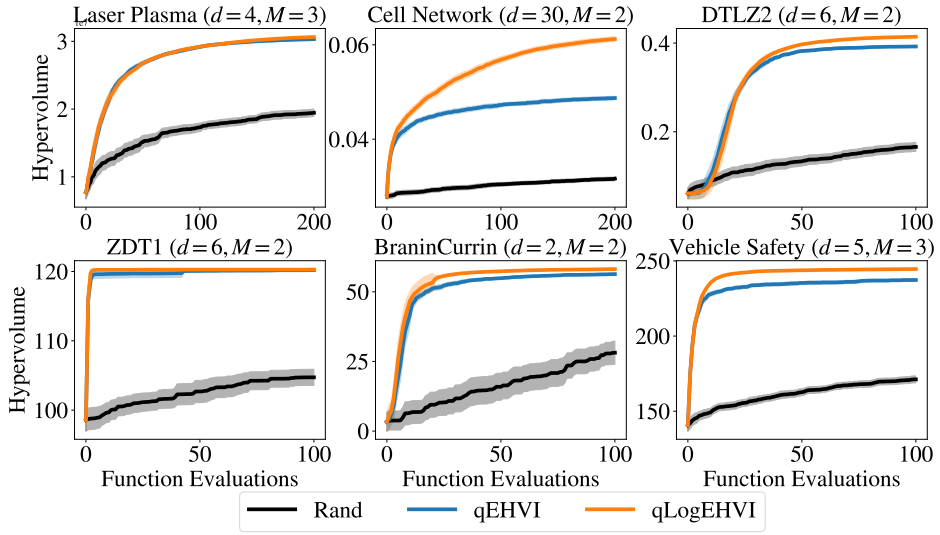

Figure 7: Sequential ($q = 1$) optimization performance on multi-objective problems, as measured by the hypervolume of the Pareto frontier across observed points.

modified acquisition functions is that they are much less dependent on heuristic (and potentially brittle) initialization strategies. Moreover, our proposed modifications do not meaningfully increase the computational complexity of the respective original acquisition function.

While our contributions may not apply verbatim to other classes of acquisition functions, our key insights and strategies do translate and could help e.g. with improving information-based [24, 59], cost-aware [37, 50], and other types of acquisition functions that are prone to similar numerical challenges. Overall, we hope that our findings will increase awareness in the community for the importance of optimizing acquisition functions well, and for the required care that is needed regarding the involved numerics.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

[2] Maximilian Balandat, Brian Karrer, Daniel R. Jiang, Samuel Daulton, Benjamin Letham, Andrew Gordon Wilson, and Eytan Bakshy. BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization. In *Advances in Neural Information Processing Systems 33*, 2020.

[3] Ricardo Baptista and Matthias Poloczek. Bayesian optimization of combinatorial structures, 2018.

[4] Syrine Belakaria, Aryan Deshwal, and Janardhan Rao Doppa. Max-value entropy search for multi-objective bayesian optimization with constraints, 2020.

[5] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/google/jax`.

[6] Richard H Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

[7] Carlos A Coello Coello and Efrén Mezura Montes. Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. *Advanced Engineering Informatics*, 16(3):193–203, 2002.

[8] Samuel Daulton, Maximilian Balandat, and Eytan Bakshy. Differentiable expected hypervolume improvement for parallel multi-objective bayesian optimization. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 9851–9864. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/6fec24eac8f18ed793f5eaad3dd7977c-Paper.pdf`.

[9] Samuel Daulton, Xingchen Wan, David Eriksson, Maximilian Balandat, Michael A. Osborne, and Eytan Bakshy. Bayesian optimization over discrete and mixed spaces via probabilistic reparameterization. In *Advances in Neural Information Processing Systems 35*, 2022.

[10] Kalyan Deb, L. Thiele, Marco Laumanns, and Eckart Zitzler. Scalable multi-objective optimization test problems. volume 1, pages 825–830, 06 2002. ISBN 0-7803-7282-4. doi: 10.1109/CEC.2002.1007032.

[11] Aryan Deshwal, Sebastian Ament, Maximilian Balandat, Eytan Bakshy, Janardhan Rao Doppa, and David Eriksson. Bayesian optimization over high-dimensional combinatorial spaces via dictionary-based embeddings. In Francisco Ruiz, Jennifer Dy, and Jan-Willem van de Meent, editors, *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pages 7021–7039. PMLR, 25–27 Apr 2023.

[12] Ryan M. Dreifuerst, Samuel Daulton, Yuchen Qian, Paul Varkey, Maximilian Balandat, Sanjay Kasturia, Anoop Tomar, Ali Yazdan, Vish Ponnampalam, and Robert W. Heath. Optimizing coverage and capacity in cellular networks using machine learning, 2021.

[13] M. T. M. Emmerich, K. C. Giannakoglou, and B. Naujoks. Single- and multiobjective evolutionary optimization assisted by gaussian random field metamodels. *IEEE Transactions on Evolutionary Computation*, 10(4):421–439, 2006.

[14] David Eriksson, Michael Pearce, Jacob Gardner, Ryan D Turner, and Matthias Poloczek. Scalable global optimization via local Bayesian optimization. In *Advances in Neural Information Processing Systems 32*, NeurIPS, 2019.

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

[16] Tobias Friedrich, Karl Bringmann, Thomas Voß, and Christian Igel. The logarithmic hypervolume indicator. In *Proceedings of the 11th workshop proceedings on Foundations of genetic algorithms*, pages 81–92, 2011.

[17] Jacob Gardner, Matt Kusner, Zhixiang, Kilian Weinberger, and John Cunningham. Bayesian optimization with inequality constraints. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 937–945, Beijing, China, 22–24 Jun 2014. PMLR.

[18] Roman Garnett. *Bayesian Optimization*. Cambridge University Press, 2023. to appear.

[19] Michael A. Gelbart, Jasper Snoek, and Ryan P. Adams. Bayesian optimization with unknown constraints. In *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence*, UAI, 2014.

[20] David Ginsbourger, Rodolphe Le Riche, and Laurent Carraro. A Multi-points Criterion for Deterministic Parallel Global Optimization based on Gaussian Processes. Technical report, March 2008. URL `https://hal.science/hal-00260579`.

[21] Robert B Gramacy, Annie Sauer, and Nathan Wycoff. Triangulation candidates for bayesian optimization. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 35933–35945. Curran Associates, Inc., 2022.

[22] Nikolaus Hansen, Sibylle D. Müller, and Petros Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). *Evolutionary Computation*, 11(1):1–18, 2003. doi: 10.1162/106365603321828970.

[23] Tim Head, Manoj Kumar, Holger Nahrstaedt, Gilles Louppe, and Iaroslav Shcherbatyi. scikit-optimize/scikit-optimize, October 2021. URL `https://doi.org/10.5281/zenodo.5565057`.

[24] José Miguel Henrández-Lobato, Matthew W. Hoffman, and Zoubin Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'14, pages 918–926, Cambridge, MA, USA, 2014. MIT Press.

[25] Carl Hvarfner, Frank Hutter, and Luigi Nardi. Joint entropy search for maximally-informed bayesian optimization. In *Advances in Neural Information Processing Systems 35*, 2022.

[26] F. Irshad, S. Karsch, and A. Döpp. Multi-objective and multi-fidelity bayesian optimization of laser-plasma acceleration. *Phys. Rev. Res.*, 5:013063, Jan 2023. doi: 10.1103/PhysRevResearch.5.013063. URL `https://link.aps.org/doi/10.1103/PhysRevResearch.5.013063`.

[27] Faran Irshad, Stefan Karsch, and Andreas Doepp. Reference dataset of multi-objective and multi- fidelity optimization in laser-plasma acceleration, January 2023. URL `https://doi.org/10.5281/zenodo.7565882`.

[28] Shali Jiang, Daniel Jiang, Maximilian Balandat, Brian Karrer, Jacob Gardner, and Roman Garnett. Efficient nonmyopic bayesian optimization via one-shot multi-step trees. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 18039–18049. Curran Associates, Inc., 2020.

[29] Donald Jones, C. Perttunen, and B. Stuckman. Lipschitzian optimisation without the lipschitz constant. *Journal of Optimization Theory and Applications*, 79:157–181, 01 1993. doi: 10.1007/BF00941892.

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

[31] Kirthevasan Kandasamy, Karun Raju Vysyaraju, Willie Neiswanger, Biswajit Paria, Christopher R. Collins, Jeff Schneider, Barnabás Póczos, and Eric P. Xing. Tuning hyperparameters without grad students: Scalable and robust bayesian optimisation with dragonfly. *J. Mach. Learn. Res.*, 21(1), jan 2020.

[32] Jungtaek Kim, Seungjin Choi, and Minsu Cho. Combinatorial bayesian optimization with random mapping functions to convex polytopes. In James Cussens and Kun Zhang, editors, *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, pages 1001–1011. PMLR, 01–05 Aug 2022.

[33] Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. *arXiv e-prints*, page arXiv:1312.6114, Dec 2013.

[34] Dieter Kraft. A software package for sequential quadratic programming. *Forschungsbericht-Deutsche Forschungs- und Versuchsanstalt fur Luft- und Raumfahrt*, 1988.

[35] H. J. Kushner. 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, 03 1964.

[36] Armin Lederer, Jonas Umlauft, and Sandra Hirche. Posterior variance analysis of gaussian processes with application to average learning curves. *arXiv preprint arXiv:1906.01404*, 2019.

[37] Eric Hans Lee, Valerio Perrone, Cedric Archambeau, and Matthias Seeger. Cost-aware Bayesian Optimization. *arXiv e-prints*, page arXiv:2003.10870, March 2020.

[38] Benjamin Letham, Brian Karrer, Guilherme Ottoni, and Eytan Bakshy. Constrained bayesian optimization with noisy experiments. *Bayesian Analysis*, 14(2):495–519, 06 2019. doi: 10.1214/18-BA1110.

[39] Qiaohao Liang, Aldair E. Gongora, Zekun Ren, Armi Tiihonen, Zhe Liu, Shijing Sun, James R. Deneault, Daniil Bash, Flore Mekki-Berrada, Saif A. Khan, Kedar Hippalgaonkar, Benji Maruyama, Keith A. Brown, John Fisher III, and Tonio Buonassisi. Benchmarking the performance of bayesian optimization across multiple experimental materials science domains. *npj Computational Materials*, 7(1):188, 2021.

[40] Xingtao Liao, Qing Li, Xujing Yang, Weigang Zhang, and Wei Li. Multiobjective optimization for crash safety design of vehicles using stepwise regression model. *Structural and Multidisciplinary Optimization*, 35:561–569, 06 2008. doi: 10.1007/s00158-007-0163-x.

[41] Mark McLeod, Stephen Roberts, and Michael A. Osborne. Optimization, fast and slow: optimally switching between local and Bayesian optimization. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3443–3452. PMLR, 10–15 Jul 2018.

[42] Jonas Močkus. On bayesian methods for seeking the extremum. In *Optimization Techniques IFIP Technical Conference: Novosibirsk, July 1–7, 1974*, pages 400–404. Springer, 1975.

[43] Jonas Mockus. The application of bayesian methods for seeking the extremum. *Towards global optimization*, 2:117–129, 1978.

[44] Hossein Mohammadi, Rodolphe Le Riche, and Eric Touboul. Making ego and cma-es complementary for global optimization. In Clarisse Dhaenens, Laetitia Jourdan, and Marie-Eléonore Marmion, editors, *Learning and Intelligent Optimization*, pages 287–292, Cham, 2015. Springer International Publishing.

[45] Henry B. Moss, David S. Leslie, Javier González, and Paul Rayson. Gibbon: General-purpose information-based bayesian optimisation. *J. Mach. Learn. Res.*, 22(1), jan 2021.

[46] Changyong Oh, Jakub Tomczak, Efstratios Gavves, and Max Welling. Combinatorial bayesian optimization using the graph cartesian product. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 2914–2924. Curran Associates, Inc., 2019.

[47] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[48] Victor Picheny, Joel Berkeley, Henry B. Moss, Hrvoje Stojic, Uri Granta, Sebastian W. Ober, Artem Artemev, Khurram Ghani, Alexander Goodall, Andrei Paleyes, Sattar Vakili, Sergio Pascual-Diaz, Stratis Markou, Jixiang Qing, Nasrulloh R. B. S Loka, and Ivo Couckuyt. Trieste: Efficiently exploring the depths of black-box functions with tensorflow, 2023. URL `https://arxiv.org/abs/2302.08436`.

[49] Carl Edward Rasmussen. *Gaussian Processes in Machine Learning*, pages 63–71. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[50] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959, 2012.

[51] S. Surjanovic and D. Bingham. Virtual library of simulation experiments: Test functions and datasets. Retrieved May 14, 2023, from `http://www.sfu.ca/~ssurjano`.

[52] Ryoji Tanabe and Hisao Ishibuchi. An easy-to-use real-world multi-objective optimization problem suite. *Applied Soft Computing*, 89:106078, 2020. ISSN 1568-4946. doi: https://doi.org/10.1016/j.asoc.2020.106078.

[53] The GPyOpt authors. GPyOpt: A bayesian optimization framework in python. `http://github.com/SheffieldML/GPyOpt`, 2016.

[54] Aimo Törn and Antanas Zilinskas. *Global optimization*, volume 350. Springer, 1989.

[55] Ben Tu, Axel Gandy, Nikolas Kantas, and Behrang Shafei. Joint entropy search for multi-objective bayesian optimization. In *Advances in Neural Information Processing Systems 35*, 2022.

[56] Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106 (1):25–57, 2006.

[57] Xingchen Wan, Vu Nguyen, Huong Ha, Binxin Ru, Cong Lu, and Michael A. Osborne. Think global and act local: Bayesian optimisation over high-dimensional categorical and mixed search spaces. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 10663–10674. PMLR, 18–24 Jul 2021.

[58] Jialei Wang, Scott C. Clark, Eric Liu, and Peter I. Frazier. Parallel bayesian global optimization of expensive functions, 2016.

[59] Zi Wang and Stefanie Jegelka. Max-value Entropy Search for Efficient Bayesian Optimization. *ArXiv e-prints*, page arXiv:1703.01968, March 2017.

[60] James Wilson, Frank Hutter, and Marc Deisenroth. Maximizing acquisition functions for bayesian optimization. In *Advances in Neural Information Processing Systems 31*, pages 9905–9916. 2018.

[61] Jian Wu and Peter I. Frazier. The parallel knowledge gradient method for batch bayesian optimization. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 3134–3142, Red Hook, NY, USA, 2016. Curran Associates Inc. ISBN 9781510838819.

[62] Jian Wu, Matthias Poloczek, Andrew Gordon Wilson, and Peter I Frazier. Bayesian optimization with gradients. In *Advances in Neural Information Processing Systems*, pages 5267–5278, 2017.

[63] Kaifeng Yang, Michael Emmerich, André Deutz, and Thomas Bäck. Multi-objective bayesian global optimization using expected hypervolume improvement gradient. *Swarm and Evolutionary Computation*, 44:945 – 956, 2019.

[64] Dawei Zhan and Huanlai Xing. Expected improvement for expensive optimization: a review. *Journal of Global Optimization*, 78(3):507–544, 2020.

[65] Jiayu Zhao, Renyu Yang, Shenghao Qiu, and Zheng Wang. Enhancing high-dimensional bayesian optimization by optimizing the acquisition function maximizer initialization. *arXiv preprint arXiv:2302.08298*, 2023.

[66] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evol. Comput.*, 8(2):173–195, jun 2000. ISSN 1063-6560. doi: 10.1162/106365600568202. URL `https://doi.org/10.1162/106365600568202`.

A Acquisition Function Details

A.1 Analytic Expected Improvement

Recall that the main challenge with computing analytic LogEI is to accurately compute $\log h$, where $h(z) = \phi(z) + z\Phi(z)$, with $\phi(z) = \exp(-z^2/2)/\sqrt{2\pi}$ and $\Phi(z) = \int_{-\infty}^{z} \phi(u)du$. To express $\log h$ in a numerically stable form as z becomes increasingly negative, we first take the log and multiply ϕ out of the argument to the logarithm:

$$\log h(z) = z^2/2 - \log(2\pi)/2 + \log\left(1 + z\frac{\Phi(z)}{\phi(z)}\right). \tag{12}$$

Fortunately, this form exposes the quadratic factor, and $\Phi(z)/\phi(z)$ can be computed via standard implementations of the scaled complementary error function `erfcx`. However, even `erfcx` can give rise to numerical underflow (though after a significantly larger range than the original formulation). To further increase numerical stability, we instead compute the $\log(\Phi(z)/\phi(z))$ using

$$\texttt{logerfcx}(x) = \begin{cases} \texttt{log}(\texttt{erfc}(x)) + x^2 & x < 0 \\ \texttt{log}(\texttt{erfcx}(x)) & x \geq 0 \end{cases} \tag{13}$$

and afterward use

$$\texttt{log1mexp}(x) = \begin{cases} \texttt{log}(-\texttt{expm1}(x)) & -\log 2 < x \\ \texttt{log1p}(-\texttt{exp}(x)) & -\log 2 \geq x \end{cases} \tag{14}$$

to compute the last term of $\log h$ in Eq. (12). In particular, we compute

$$\texttt{log\_h}(z) = \begin{cases} \texttt{log}(\phi(z) + z\Phi(z)) & z > -1 \\ -z^2/2 - c_1 + \texttt{log1mexp}(\texttt{logerfcx}(-z/\sqrt{2}) + |z| + c_2) & z < -1 \end{cases} \tag{15}$$

where $c_1 = \log(2\pi)/2$, and $c_2 = \log(\pi/2)/2$.

Figure 8 shows both the numerical failure mode of a naïve implementation of EI, which becomes *exactly* zero numerically for moderately small z, while the evaluation via `log_h` in Eq. (15) exhibits quadratic asymptotic behavior that is particularly amenable to numerical optimization routines.

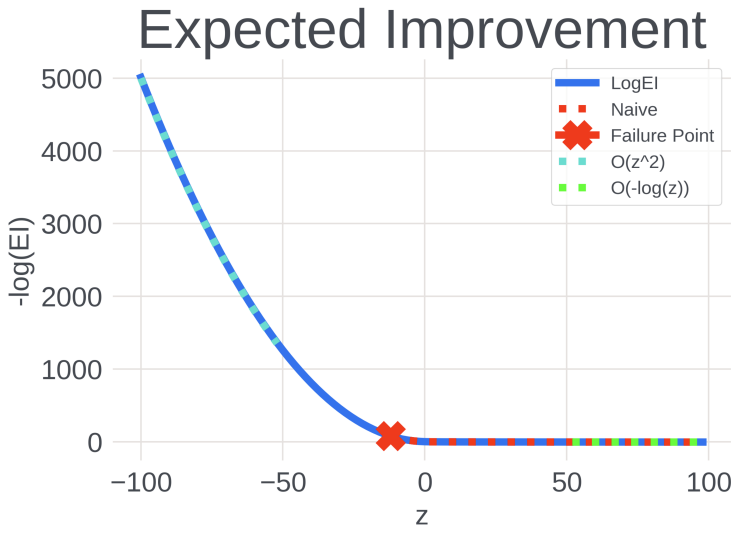

Figure 8: Plot of the $\log h$, computed via $\texttt{log} \circ h$ and `log_h` in Eq. (15). Crucially, the naïve implementation fails as $z = (\mu(x) - f^*)/\sigma(x)$ becomes increasingly negative, due to being exactly numerically zero, while our proposed implementation exhibits quadratic asymptotic behavior.

A.2 Monte-Carlo Expected Improvement

For Monte-Carlo, we cannot directly apply similar numerical improvements as for the analytical version, because the utility values (integrand of Eq. (4)) on the sample level are likely to be *mathematically* zero. For this reason, we first smoothly approximate the acquisition utility and subsequently apply log transformations to the approximate acquisition function.

To this end, a natural choice is $\mathrm{softplus}_{\tau_0}(x) = \tau_0 \log(1 + \exp(x/\tau_0))$ for smoothing the $\max(0, x)$, where τ_0 is a temperature parameter governing the approximation error. Further, we approximate the $\max_i$ over the q candidates by the norm $\|\cdot\|_{1/\tau_{\max}}$ and note that the approximation error introduced by both smooth approximations can be bound tightly as a function of two "temperature" parameters τ_0 and $\tau_{\max}$, see Lemma 2.

Importantly, the smoothing alone only solves the problem of having mathematically zero gradients, not that of having numerically vanishing gradients, as we have shown for the analytical case above. For this reason, we transform all smoothed computations to log space and thus need the following special implementation of $\log \circ \mathrm{softplus}$ that can be evaluated stably for a very large range of inputs:

$$\texttt{logsoftplus}_\tau(x) = \begin{cases} [\texttt{log} \circ \texttt{softplus}_\tau](x) & x/\tau > l \\ x/\tau + \texttt{log}(\tau) & x/\tau \leq l \end{cases}$$

where τ is a temperature parameter and l depends on the floating point precision used, around -35 for double precision in our implementation.

Note that the lower branch of `logsoftplus` is approximate. Using a Taylor expansion of $\log(1+z) = z - z^2/2 + \mathcal{O}(z^3)$ around $z = 0$, we can see that the approximation error is $\mathcal{O}(z^2)$, and therefore, $\log(\log(1 + \exp(x))) = x + \mathcal{O}(\exp(x)^2)$, which converges to x exponentially quickly. l is chosen in our implementation so that no significant digit is lost in dropping the second order term from the lower branch.

Having defined `logsoftplus`, we further note that

$$\begin{aligned} \log \|\mathbf{x}\|_{1/\tau_{\max}} &= \log \left(\sum_i x_i^{1/\tau_{\max}} \right)^{\tau_{\max}} \\ &= \tau_{\max} \log \left(\sum_i \exp(\log(x_i)/\tau_{\max}) \right) \\ &= \tau_{\max} \texttt{logsumexp}_i \left(\log(x_i)/\tau_{\max} \right) \end{aligned}$$

Therefore, we can express the logarithm of the smoothed acquisition utility per sample as

$$\tau_{\max} \texttt{logsumexp}_i(\texttt{logsoftplus}_{\tau_0}(z_i)/\tau_{\max})$$

Applying another `logsumexp` to compute the logarithm of the mean of acquisition utilities over a set of Monte Carlo samples gives rise to the expression in Eq. (10).

In particular for large batches (large q), this expression can still give rise to vanishing gradients for some candidates, which is due to the large dynamic range of the outputs of the `logsoftplus` when $x << 0$. To solve this problem, we propose a new class of smooth approximations to the "hard" non-linearities that decay as $\mathcal{O}(1/x^2)$ as $x \to -\infty$ in the next section.

A.3 A Class of Smooth Approximations with Fat Tails for Larger Batches

A regular $\mathrm{softplus}(x) = \log(1 + \exp(x))$ function smoothly approximates the ReLU non-linearity and – in conjunction with the log transformations – is sufficient to achieve good numerical behavior for small batches of the Monte Carlo acquisition functions. However, as more candidates are added, $\log \mathrm{softplus}(x) = \log(\log(1 + \exp(x)))$ is increasingly likely to have a high dynamic range as for $x \ll 0$, $\log \mathrm{softplus}_\tau(x) \sim -x/\tau$. If $\tau > 0$ is chosen to be small, $(-x/\tau)$ can vary orders of magnitude within a single batch. This becomes problematic when we approximate the maximum utility over the batch of candidates, since `logsumexp` only propagates numerically non-zero gradients to inputs that are no smaller than approximately $(\max_j x_j - 700)$ in double precision, another source of vanishing gradients.

To solve this problem, we propose a new smooth approximation to the ReLU, maximum, and indicator functions that decay only polynomially as $x \to -\infty$, instead of exponentially, like the canonical softplus. The high level idea is to use $(1 + x^2)^{-1}$, which is proportional to the Cauchy density function (and is also known as a Lorentzian), in ways that maintain key properties of existing smooth approximations – convexity, positivity, etc – while changing the asymptotic behavior of the functions from exponential to $\mathcal{O}(1/x^2)$ as $x \to -\infty$, also known as a "fat tail". Further, we will show that the proposed smooth approximations satisfy similar maximum error bounds as their exponentially decaying counterparts, thereby permitting a similar approximation guarantee as Lemma 2 with minor adjusments to the involved constants.

Fat Softplus We define

$$\varphi_+(x) = \alpha(1 + x^2)^{-1} + \log(1 + \exp(x)), \tag{16}$$

for a positive scalar α. The following result shows that we can ensure the monotonicity and convexity – both important properties of the ReLU that we would like to maintain in our approximation – of g by carefully choosing α.

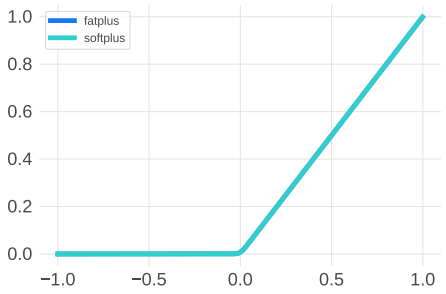

Figure 9: The fat softplus approximates $\max(x, 0)$ similarly tightly as the regular softplus and is also monotonic, convex, and positive. The plot used a temperature of $\tau_0 = 0.01$.

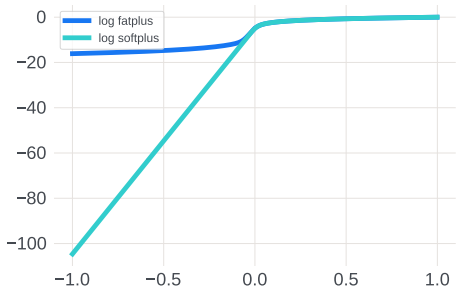

Figure 10: The fat softplus has an $\mathcal{O}(1/x^2)$ asymptotic decay, versus the $\mathcal{O}(\exp(x))$ decay as $x \to -\infty$, moderating the dynamic range of the quantities involved in parallel LogEI.

Lemma 3 (Monotonicity and Convexity)**.** *$\varphi_+(x)$ is positive, monotonically increasing, and strictly convex for α satisfying*

$$0 \leq \alpha < \frac{e^{1/\sqrt{3}}}{2\left(1 + e^{1/\sqrt{3}}\right)}.$$

Proof. Positivity follows due to $\alpha \geq 0$, and both sumands being positive. Monotonicity and convexity can be shown via canonical differential calculus and bounding relevant quantities.

In particular, regarding monotonicity, we want to select α so that the first derivative is bounded below by zero:

$$\partial_x \varphi_+(x) = \frac{e^x}{1 + e^x} - \alpha \frac{2x}{(1 + x^2)^2}$$

First, we note that $\partial_x \varphi_+(x)$ is positive for $x < 0$ and any α, since both terms are positive in this regime. For $x \geq 0$, $\frac{e^x}{1+e^x} = (1 + e^{-x})^{-1} \geq 1/2$, and $-1/(1 + x^2)^2 \geq -1/(1 + x^2)$, so that

$$\partial_x \varphi_+(x) \geq \frac{1}{2} - \alpha \frac{2x}{(1 + x^2)}$$

Forcing $\frac{1}{2} - \alpha \frac{2x}{(1+x^2)} > 0$, and multiplying by $(1 + x^2)^2$ gives rise to a quadratic equation whose roots are $x = 2\alpha \pm \sqrt{4\alpha^2 - 1}$. Thus, there are no real roots for $\alpha < 1/2$. Since the derivative is certainly positive for the negative reals and the guaranteed non-existence of roots implies that the derivative cannot cross zero elsewhere, $0 \leq \alpha < 1/2$ is a sufficient condition for monotonicity of φ_+.

Regarding convexity, our goal is to prove a similar condition on α that guarantees the positivity of the second derivative:

$$\partial_x^2 \varphi_+(x) = \alpha \frac{6x^2 - 2}{(1 + x^2)^3} + \frac{e^{-x}}{(1 + e^{-x})^2}$$

Note that $\frac{6x^2-2}{(1+x^2)^3}$ is symmetric around 0, is negative in $(-\sqrt{1/3}, \sqrt{1/3})$ and has a minimum of -2 at 0. $\frac{e^{-x}}{(1+e^{-x})^2}$ is symmetric around zero and decreasing away from zero. Since the rational polynomial is only negative in $(-\sqrt{1/3}, \sqrt{1/3})$, we can lower bound $\frac{e^{-x}}{(1+e^{-x})^2} > \frac{e^{-\sqrt{1/3}}}{(1+e^{-\sqrt{1/3}})^2}$ in $(-\sqrt{1/3}, \sqrt{1/3})$. Therefore,

$$\partial_x^2 \varphi_+(x) \geq \frac{e^{-x}}{(1 + e^{-x})^2} - 2\alpha$$

Forcing $\frac{e^{-\sqrt{1/3}}}{(1+e^{-\sqrt{1/3}})^2} - 2\alpha > 0$ and rearranging yields the result. Since $\frac{e^{-\sqrt{1/3}}}{(1+e^{-\sqrt{1/3}})^2}/2 \sim 0.115135$, the convexity condition is stronger than the monotonicity condition and therefore subsumes it. □

Importantly φ decays only polynomially for increasingly negative inputs, and therefore $\log \varphi$ only logarithmically, which keeps the range of φ constrained to values that are more manageable numerically. Similar to Lemma 5, one can show that

$$|\tau \varphi_+(x/\tau) - \texttt{ReLU}(x)| \leq (\alpha + \log(2))\, \tau. \tag{17}$$

There are a large number of approximations or variants of the ReLU that have been proposed as activation functions of artificial neural networks, but to our knowledge, none satisfy the properties that we seek here: (1) smoothness, (2) positivity, (3) monotonicity, (4) convexity, and (5) polynomial decay. For example, the leaky ReLU does not satisfy (1) and (2), and the ELU does not satisfy (5).

Fat Maximum The canonical `logsumexp` approximation to $\max_i x_i$ suffers from numerically vanishing gradients if $\max_i x_i - \min_j x_j$ is larger a moderate threshold, around 760 in double precision, depending on the floating point implementation. In particular, while elements close to the maximum receive numerically non-zero gradients, elements far away are increasingly likely to have a numerically zero gradient. To fix this behavior for the smooth maximum approximation, we propose

$$\varphi_{\max}(\mathbf{x}) = \max_j x_j + \tau \log \sum_i \left[1 + \left(\frac{x_i - \max_j x_j}{\tau} \right)^2 \right]^{-1}. \tag{18}$$

This approximation to the maximum has the same error bound to the true maximum as the `logsumexp` approximation:

Lemma 4. *Given* $\tau > 0$

$$\max_i x_i \leq \tau \phi_{\max}(x/\tau) \leq \max_i x_i + \tau \log(d). \tag{19}$$

Proof. Regarding the lower bound, by definition there is an index i such that $x_i = \max_j x_j$. For this index, the associated summand in (18) is 1. Since all sumands are positive, the entire sum is lower bounded by 1, hence

$$\tau \log \sum_i \left[1 + \left(\frac{x_i - \max_j x_j}{\tau} \right)^2 \right]^{-1} > \tau \log(1) = 0$$

Adding $\max_j x_j$ to the inequality finishes the proof for the lower bound.

Regarding the upper bound, (18) can be maximized when $x_i = \max_j x_j$ for all i, in which case each $(x_i - \max_j x_j)^2$ is minimized, and hence each summand is maximized. In this case,

$$\tau \log \sum_i \left[1 + \left(\frac{x_i - \max_j x_j}{\tau} \right)^2 \right]^{-1} \leq \tau \log \left(\sum_i 1 \right) = \tau \log(d).$$

Adding $\max_j x_j$ to the inequality finishes the proof for the upper bound. □

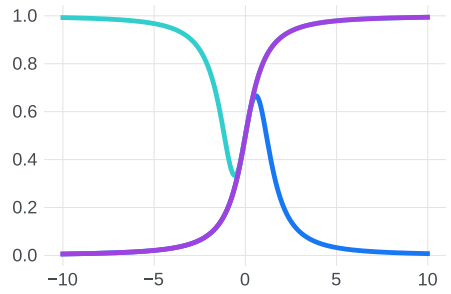

Figure 11: We construct the fat sigmoid approximation (purple) by splicing together two Lorentzians (blue and teal) at one of their inflection points.

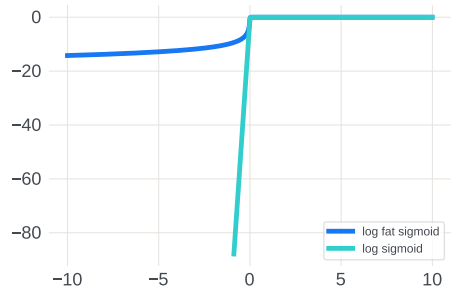

Figure 12: The fat sigmoid approximation ($\tau = 0.01$) decays as $\mathcal{O}(1/x^2)$ instead of $\mathcal{O}(\exp(x))$ as $x \to -\infty$, minimizing the dynamic range of the numerical quantities in constrained qLogEI.

Fat Sigmoid Notably, we encountered a similar problem with using regular (log)-sigmoids to smooth the constraint indicators for EI with black-box constraints. Here, we want the smooth approximation ι to satisfy 1) positivity, 2) monotonicity, 3) polynomial decay, and 4) $\iota(x) = 1/2 - \iota(-x)$. Let $\gamma = \sqrt{1/3}$, then we define

$$\iota(x) = \begin{cases} \frac{2}{3}\left(1 + (x-\gamma)^2\right)^{-1} & x < 0, \\ 1 - \frac{2}{3}\left(1 + (x+\gamma)^2\right)^{-1} & x \geq 0. \end{cases}$$

ι is monotonically increasing, satisfies $\iota(x) \to 1$ as $x \to \infty$, $\iota(0) = 1/2$, and $\iota(x) = \mathcal{O}(1/x^2)$ as $x \to -\infty$. Further, we note that the asymptotics are primarily important here, but that we can also make the approximation tighter by introducing a temperature parameter τ, and letting $\iota_\tau(x) = \iota(x/\tau)$. The approximation error of $\iota_\tau(x)$ to the Heaviside step function becomes tighter point-wise as $\tau \to 0+$, except for at the origin where $\iota_\tau(x) = 1/2$, similar to the canonical sigmoid.

A.4 Constrained Expected Improvement

For the analytical case, many computational frameworks already provide a numerically stable implementation of the logarithm of the Gaussian cummulative distribution function, in the case of PyTorch, `torch.special.log_ndtr`, which can be readily used in conjunction with our implementation of `LogEI`, as described in Sec. 4.3.

For the case of Monte-Carlo parallel EI, we implemented the fat-tailed ι function from Sec. A.3 to approximate the constraint indicator and compute the per-candidate, per-sample acquisition utility using

$$(\texttt{logsoftplus}_{\tau_0}(\xi_i(x_j) - y^*) + \sum_k \texttt{log} \circ \iota\left(-\frac{\xi_i^{(k)}}{\tau_{\text{cons}}}\right),$$

where $\xi_i^{(k)}$ is the ith sample of the kth constraint model, and τ_{cons} is the temperature parameter controlling the approximation to the constraint indicator. While this functionality is in our implementation, our benchmark results use the analytical version.

A.5 Parallel Expected Hypervolume Improvement

The hypervolume improvement can be computed via the inclusion-exclusion principle, see [8] for details, we focus on the numerical issues concerning qEHVI here. To this end, we define

$$z^{(m)}_{k,i_1,\dots,i_j} = \min\left[\mathbf{u}_k, \mathbf{f}(\mathbf{x}_{i_1}), \dots, \mathbf{f}(\mathbf{x}_{i_j})\right],$$

where $\mathbf{f}$ is the vector-valued objective function, and $\mathbf{u}_k$ is the vector of upper bounds of one of K hyper-rectangles that partition the non-Pareto-dominated space, see [8] for details on the partitioning.

Letting $\mathbf{l}_k$ be the corresponding lower bounds of the hyper-rectangles, the hypervolume improvement can then be computed as

$$\text{HVI}(\{\mathbf{f}(\mathbf{x}_i)\}_{i=1}^q = \sum_{k=1}^{K} \sum_{j=1}^{q} \sum_{X_j \in \mathcal{X}_j} (-1)^{j+1} \prod_{m=1}^{M} [z_{k,X_j}^{(m)} - l_k^{(m)}]_+, \tag{20}$$

where $\mathcal{X}_j = \{X_j \subset \mathcal{X}_{\text{cand}} : |X_j| = j\}$ is the superset of all subsets of $\mathcal{X}_{\text{cand}}$ of size j and $z_{k,X_j}^{(m)} = z_{k,i_1,\dots,i_j}^{(m)}$ for $X_j = \{\mathbf{x}_{i_1}, \dots, \mathbf{x}_{i_j}\}$.

To find a numerically stable formulation of the logarithm of this expression, we first re-purpose the $\varphi_{\max}$ function to compute the minimum in the expression of $z_{k,i_1,\dots,i_j}^{(m)}$, like so $\varphi_{\min}(x) = -\varphi_{\max}(-x)$. Further, we use the φ_+ function of Sec. A.3 as for the single objective case to approximate $[z_{k,X_j}^{(m)} - l_k^{(m)}]_+$. We then have

$$\log \prod_{m=1}^{M} \varphi_+[z_{k,X_j}^{(m)} - l_k^{(m)}] = \sum_{m=1}^{M} \log \varphi_+[z_{k,X_j}^{(m)} - l_k^{(m)}] \tag{21}$$

Since we can only transform positive quantities to log space, we split the sum in Eq. (20) into positive and negative components, depending on the sign of $(-1)^{j+1}$, and compute the result using a numerically stable implementation of $\log(\exp(\text{log of positive terms}) - \exp(\text{log of negative terms})$. The remaining sums over k and q can be carried out by applying `logsumexp` to the resulting quantity. Finally, applying `logsumexp` to reduce over an additional Monte-Carlo sample dimension yields the formulation of qLogEHVI that we use in our multi-objective benchmarks.

B Strategies for Optimizing Acquisition Functions

As discussed in Section 2.3, a variety of different approaches and heuristics have been applied to the problem of optimizing acquisition functions. For the purpose of this work, we only consider continuous domains $\mathbb{X}$. While discrete and/or mixed domains are also relevant in practice and have received substantial attention in recent years (see e.g. Baptista and Poloczek [3], Daulton et al. [9], Deshwal et al. [11], Kim et al. [32], Oh et al. [46], Wan et al. [57]), our work here on improving acquisition functions is largely orthogonal to this (though the largest gains should be expected when using gradient-based optimizers, as is done in mixed-variable BO when conditioning on discrete variables, or when performing discrete or mixed BO using continuous relaxations, probabilistic reparameterization, or straight-through estimators [9]).

Arguably the simplest approach to optimizing acquisition functions is by grid search or random search. While variants of this combined with local descent can make sense in the context of optimizing over discrete or mixed spaces and when acquisition functions can be evaluated efficiently in batch (e.g. on GPUs), this clearly does not scale to higher-dimensional continuous domains due to the exponential growth of space to cover.

Another relatively straightforward approach is to use zeroth-order methods such as `DIRECT` [29] (used e.g. by `Dragonfly` [31]) or the popular CMA-ES [22]. These approaches are easy implement as they avoid the need to compute gradients of acquisition functions. However, not relying on gradients is also what renders their optimization performance inferior to gradient based methods, especially for higher-dimensional problems and/or joint batch optimization in parallel Bayesian optimization.

The most common approach to optimizing acquisition functions on continuous domains is using gradient descent-type algorithms. Gradients are either be computed based on analytically derived closed-form expressions, or via auto-differentiation capabilities of modern ML systems such as PyTorch [47], Tensorflow [1], or JAX [5].

For analytic acquisition functions, a common choice of optimizer is L-BFGS-B [6], a quasi-second order method that uses gradient information to approximate the Hessian and supports box constraints. If other, more general constraints are imposed on the domain, other general purpose nonlinear optimizers such as `SLSQP` [34] or `IPOPT` [56] are used (e.g. by `BoTorch`). For Monte Carlo (MC) acquisition functions, Wilson et al. [60] proposes using stochastic gradient ascent (SGA) based on stochastic gradient estimates obtained via the reparameterization trick [33]. Stochastic first-order

algorithms are also used by others, including e.g. Wang et al. [58] and Daulton et al. [9]. Balandat et al. [2] build on the work by Wilson et al. [60] and show how sample average approximation (SAA) can be employed to obtain deterministic gradient estimates for MC acquisition functions, which has the advantage of being able to leverage the improved convergence rates of optimization algorithms designed for deterministic functions such as L-BFGS-B. This general approach has since been used for a variety of other acquisition functions, including e.g. Daulton et al. [8] and Jiang et al. [28].

Very few implementations of Bayesian Optimization actually use higher-order derivative information, as this either requires complex derivations of analytical expressions and their custom implementation, or computation of second-order derivatives via automated differentiation, which is less well supported and computationally much more costly than computing only first-order derivatives. One notable exception is `Cornell-MOE` [61, 62], which supports Newton's method (though this is limited to the acquisition functions implemented in C++ within the library and not easily extensible to other acquisition functions).

B.1 Common initialization heuristics for multi-start gradient-descent

One of the key issues to deal with gradient-based optimization in the context of optimizing acquisition functions is the optimizer getting stuck in local optima due to the generally highly non-convex objective. This is typically addressed by means of restarting the optimizer from a number of different initial conditions distributed across the domain.

A variety of different heuristics have been proposed for this. The most basic one is to restart from random points uniformly sampled from the domain (for instance, `scikit-optimize` [23] uses this strategy). However, as we have argued in this paper, acquisition functions can be (numerically) zero in large parts of the domain, and so purely random restarts can become ineffective, especially in higher dimensions and with more data points. A common strategy is therefore to either augment or bias the restart point selection to include initial conditions that are closer to "promising points". `GPyOpt` [53] augments random restarts with the best points observed so far, or alternatively points generated via Thompson sampling. `Spearmint` [50] initializes starting points based on Gaussian perturbations of the current best point. `BoTorch` [2] selects initial points by performing Boltzmann sampling on a set of random points according to their acquisition function value; the goal of this strategy is to achieve a biased random sampling across the domain that is likely to generate more points around regions with high acquisition value, but remains asymptotically space-filling. The initialization strategy used by `Trieste` [48] works similarly to the one in `BoTorch`, but instead of using soft-randomization via Boltzmann sampling, it simply selects the top-k points. Most recently, Gramacy et al. [21] proposed distributing initial conditions using a Delaunay triangulation of previously observed data points. This is an interesting approach that generalizes the idea of initializing "in between" observed points from the single-dimensional case. However, this approach does not scale well with the problem dimension and the number of observed data points due to the complexity of computing the triangulation (with wall time empirically found to be exponential in the dimension, see [21, Fig. 3] and worst-case quadratic in the number of observed points).

However, while these initialization strategies can help substantially with better optimizing acquisition functions, they ultimately cannot resolve foundational issues with acquisition functions themselves. Ensuring that acquisition functions provides enough gradient information (not just mathematically but also numerically) is therefore key to be able to optimize it effectively, especially in higher dimensions and with more observed data points.

C Proofs

Lemma 1. *Suppose f is drawn from a Gaussian process prior P_f, $y^* \leq f^*$, μ_n, σ_n are the mean and standard deviation of the posterior $P_f(f|\mathcal{D}_n)$ and $B \in \mathbb{R}$. Then with probability $1-\delta$,*

$$P_x\left(\frac{\mu_n(x) - y_n^*}{\sigma_n(x)} < B\right) \geq P_x\left(f(x) < f^* - \epsilon_n\right) \tag{7}$$

where $\epsilon_n = (f^* - y_n^*) + \left(\sqrt{-2\log(2\delta)} - B\right) \max_x \sigma_n(x)$.

Proof.

$$\frac{\mu_n(x) - y^*}{\sigma_n(x)} = \frac{\mu_n(x) - f(x)}{\sigma_n(x)} + \frac{f(x) - f^*}{\sigma_n(x)} \tag{22}$$

We proceed by bounding the first term on the right hand side. Note that by assumption, $f(x) \sim \mathcal{N}(\mu_n(x), \sigma_n(x)^2)$ and thus $(\mu_n(x) - f(x))/\ \sigma_n(x) \sim \mathcal{N}(0, 1)$. For a positive $C > 0$ then, we use a standard bound on the Gaussian tail probability to attain

$$P\left(\frac{\mu_n(x) - f(x)}{\sigma_n(x)} > C\right) \leq e^{-C^2/2}/2. \tag{23}$$

Therefore, $(\mu(x) - f(x))/\sigma_n(x) < C$ with probability $1 - \delta$ if $C = \sqrt{-2\log(2\delta)}$.

Using the bound just derived, and forcing the resulting upper bound to be less than B yields a sufficient condition to imply $\mu_n(x) - y_n^* < B\sigma_n(x)$:

$$\frac{\mu_n(x) - y^*}{\sigma_n(x)} \leq C + \frac{f(x) - y^*}{\sigma_n(x)} < B \tag{24}$$

Re-arranging and using $y^* = f^* + (y^* - f^*)$ we get with probability $1 - \delta$,

$$f(x) \leq f^* - (f^* - y_n^*) - (\sqrt{-2\log(2\delta)} - B)\sigma_n(x). \tag{25}$$

Thus, we get

$$\begin{aligned} P_x\left(\frac{\mu_n(x) - y_n^*}{\sigma_n(x)} < B\right) &\geq P_x\left(f(x) \leq f^* - (f^* - y_n^*) - (\sqrt{-2\log(2\delta)} - B)\sigma_n(x)\right) \\ &\geq P_x\left(f(x) \leq f^* - (f^* - y_n^*) - (\sqrt{-2\log(2\delta)} - B)\max_x \sigma_n(x)\right). \end{aligned} \tag{26}$$

Note that the last inequality gives a bound that is not direclty dependent on the evaluation of the posterior statistics of the surrogate at any specific x. Rather, it is dependent on the optimality gap $f^* - y_n^*$ and the maximal posterior standard deviation, or a bound thereof. Letting $\epsilon_n = (f^* - y_n^*) - (\sqrt{-2\log(2\delta)} - B)\max_x \sigma_n(x)$ finishes the proof. □

Lemma 2. *[Approximation Guarantee] Given the temperature parameters τ_0 and $\tau_{\max}$, the approximation error of* qLogEI *to* qEI *is bounded by*

$$\left|\exp(\text{qLogEI}(x)) - \text{qEI}(x)\right| \leq \log(2)\ \tau_0 + q^{\tau_{\max}} - 1. \tag{11}$$

Proof. Let $z_{iq} = \xi(x_q) - y^*$, where $i \in \{1, ..., n\}$, and for brevity of notation, and let `lse`, `lsp` refer to the `logsumexp` and `logsoftplus` functions, respectively, and $\texttt{ReLU}(x) = [x]_+$. We then bound $n|e^{\text{qLogEI}(x)} - \text{qEI}(x)|$ by

$$\begin{aligned} &\left|\exp(\texttt{lse}_i(\tau_{\max}\texttt{lse}_q(\texttt{lsp}_{\tau_0}(z_{iq})/\tau_{\max}))) - \sum_i \max_q \texttt{ReLU}(z_{iq})\right| \\ &\leq \sum_i \left|\exp(\tau_{\max}\texttt{lse}_q(\texttt{lsp}_{\tau_0}(z_{iq})/\tau_{\max})) - \max_q \texttt{ReLU}(z_{iq})\right| \\ &= \sum_i \left|\|\texttt{softplus}_{\tau_0}(z_{i\cdot})\|_{1/\tau_{\max}} - \max_q \texttt{ReLU}(z_{iq})\right| \\ &\leq \sum_i \left|\|\texttt{softplus}_{\tau_0}(z_{i\cdot})\|_{1/\tau_{\max}} - \max_q \texttt{softplus}_{\tau_0}(z_{iq})\right| \\ &\quad + \left|\max_q \texttt{softplus}_{\tau_0}(z_{iq}) - \max_q \texttt{ReLU}(z_{iq})\right| \end{aligned} \tag{27}$$

First and second inequalities are due to the triangle inequality, where for the second we used $|a - c| \leq |a - b| + |b - c|$ with $b = \max_q \texttt{softplus}(z_{iq})$.

To bound the first term in the sum, note that $\|x\|_\infty \leq \|\mathbf{x}\|_q \leq \|x\|_\infty d^{1/q}$, thus $|\|x\|_q - \|x\|_\infty| \leq d^{1/q} - 1$, and therefore

$$\left| \|\texttt{softplus}_{\tau_0}(z_{i\cdot})\|_{1/\tau_{\max}} - \max_q \texttt{softplus}_{\tau_0}(z_{iq}) \right| \leq d^{\tau_{\max}} - 1.$$

The second term in the sum can be bound due to $|\texttt{softplus}_{\tau_0}(x) - \texttt{ReLU}(x)| \leq \log(2)\tau_0$ (see Lemma 5 below) and therefore,

$$\left| \max_q \texttt{softplus}_{\tau_0}(z_{iq}) - \max_q \texttt{ReLU}_{\tau_0}(z_{iq}) \right| \leq \log(2)\tau_0.$$

Dividing Eq. (27) by n to compute the sample mean finishes the proof for the Monte-Carlo approximations to the acquisition value. Taking $n \to \infty$ further proves the result for the mathematical definitions of the parallel acquisition values, i.e. Eq. (4). □

Approximating the ReLU using the $\texttt{softplus}_\tau(x) = \tau \log(1 + \exp(x/\tau))$ function leads to an approximation error that is at most τ in the infinity norm, i.e. $\|\texttt{softplus}_\tau - \texttt{ReLU}\|_\infty = \log(2)\tau$. The following lemma formally proves this.

Lemma 5. *Given $\tau > 0$, we have for all $x \in \mathbb{R}$,*

$$|\textit{softplus}_\tau(x) - \textit{ReLU}(x)| \leq \log(2)\ \tau. \tag{28}$$

Proof. Taking the (sub-)derivative of $\texttt{softplus}_\tau - \texttt{ReLU}$, we get

$$\partial_x \texttt{softplus}_\tau(x) - \texttt{ReLU}(x) = (1 + e^{-x/\tau})^{-1} - \begin{cases} 1 & x > 0 \\ 0 & x \leq 0 \end{cases}$$

which is positive for all $x < 0$ and negative for all $x > 0$, hence the extremum must be at x, at which point $\texttt{softplus}_\tau(0) - \texttt{ReLU}(0) = \log(2)\tau$. Analyzing the asymptotic behavior, $\lim_{x \to \pm\infty}(\texttt{softplus}_\tau(x) - \texttt{ReLU}(x)) = 0$, and therefore $\texttt{softplus}_\tau(x) > \texttt{ReLU}(x)$ for $x \in \mathbb{R}$. □

D Additional Empirical Details and Results

D.1 Experimental details

All algorithms are implemented in BoTorch. The analytic EI, qEI, cEI utilize the standard BoTorch implementations. We utilize the original authors' implementations of single objective JES [25], GIBBON [45], and multi-objective JES [55], which are all available in the main BoTorch repository. All simulations are ran with 32 replicates and error bars represent ± 2 times the standard error of the mean. We use a Matern-5/2 kernel with automatic relevance determination (ARD), i.e. separate length-scales for each input dimension, and a top-hat prior on the length-scales in $[0.01, 100]$. The input spaces are normalized to the unit hyper-cube and the objective values are standardized during each optimization iteration.

D.2 Combining LogEI with TuRBO for High-Dimensional Bayesian Optimization

In the main text, we show how LogEI performs particularly well relative to other baselines in high dimensionsional spaces. Here, we show how LogEI can work synergistically with trust-region based methods for high-dimensional BO, such as TuRBO [14].

Fig. 13 compares the performance of LogEI, TuRBO-1 + LogEI, TuRBO-1 + EI, as well as the original Thompson-sampling based implementation for the 50d Ackley test problem. Combining TuRBO-1 with LogEI results in substantially better performance than the baselines. Since we optimize batches of $q = 50$ candidates jointly, we also increase the number of Monte-Carlo samples from the Gaussian process from 128, the BoTorch default, to 512, and use the fat-tailed smooth approximations of Sec. A.3 to ensure a strong gradient signal to all candidates of the batch.

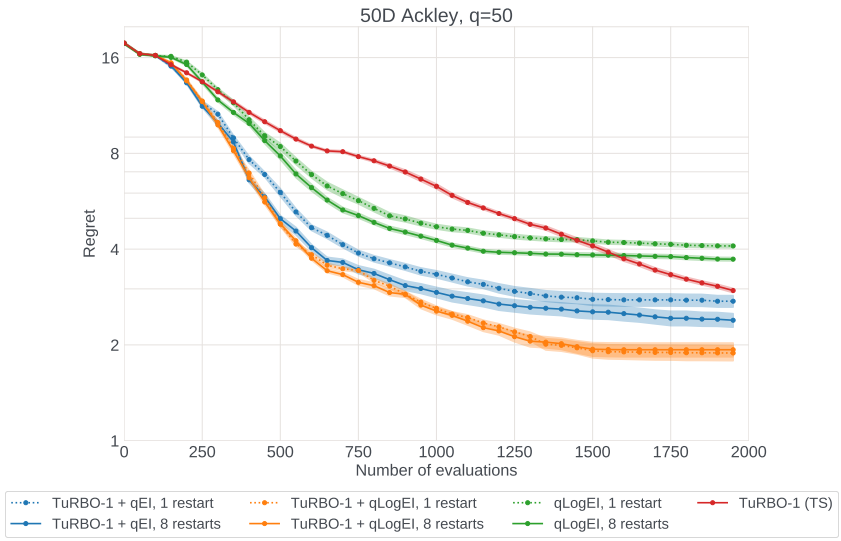

Figure 13: Combining LogEI with TuRBO on the high-dimensional on the 50d Ackley problem yields significant improvement in sample complexity. Unlike qEI, no random restarts are necessary to achieve good performance when performing joint optimization of the batch ($q = 50$). Notably, LogEI without TuRBO is comparable to the performance of TuRBO with Thompson sampling.

D.3 Constrained Problems

While running the benchmarks using cEI in section 5, we found that we in fact improved upon a best known result from the literature. We compare with the results in Coello and Montes [7], which are generated using 30 runs of **80,000 function evaluations** each.

- For the pressure vessel design problem, Coello and Montes [7] quote a best-case feasible objective of 6059.946341. Out of just 16 different runs, LogEI achieves a worst-case feasible objective of 5659.1108 **after only 110 evaluations**, and a best case of 5651.8862, a notable reduction in objective value using almost three orders of magnitude fewer function evaluations.
- For the welded beam problem, Coello and Montes [7] quote 1.728226, whereas LogEI found a best case of 1.7496 after 110 evaluations, which is lightly worse, but we stress that this is using three orders of magnitude fewer evaluations.
- For the tension-compression problem, LogEI found a feasible solution with value 0.0129 after 110 evaluations compared to the 0.012681 reported in in [7].

We emphasize that genetic algorithms and BO are generally concerned with distinct problem classes: BO focuses heavily on sample efficiency and the small-data regime, while genetic algorithms often utilize a substantially larger number of function evaluations. The results here show that in this case BO is competitive with and can even outperforms a genetic algorithm, using only a tiny fraction of the sample budget. Sample efficiency is particularly relevant for physical simulators whose evaluation takes significant computational effort, often rendering several tens of thousands of evaluations infeasible.

D.4 Parallel Bayesian Optimization with cross-batch constraints

In some parallel Bayesian optimization settings, batch optimization is subject to non-trivial constraints across the batch elements. A natural example for this are budget constraints. For instance, in the context of experimental material science, consider the case where each manufactured compound requires a certain amount of different materials (as described by its parameters), but there is only a fixed total amount of material available (e.g., because the stock is limited due to cost and/or storage capacity). In such a situation, batch generation will be subject to a budget constraint that is not separable across the elements of the batch. Importantly, in that case sequential greedy batch

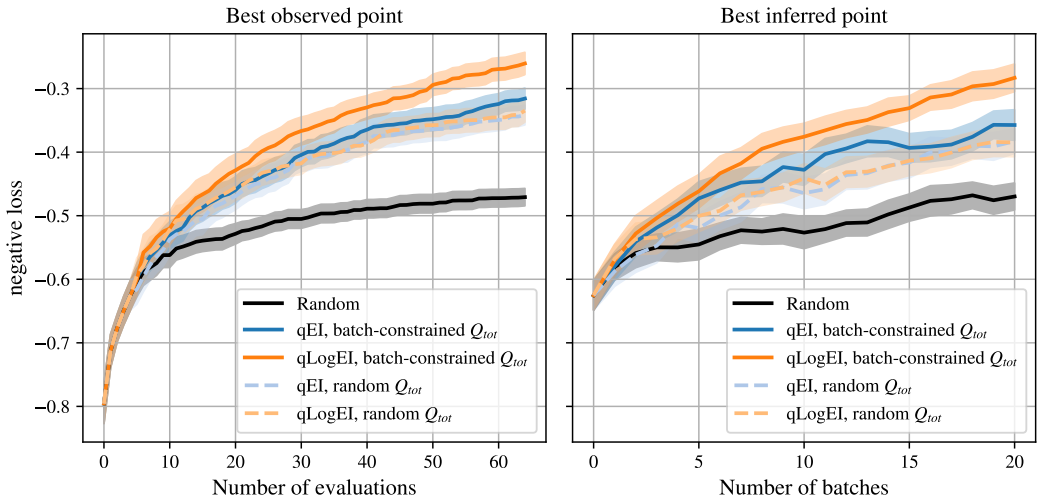

Figure 14: Optimization results on the nanomaterial synthesis material science problem with cross-batch constraints. While qLogEI outperforms qEI under the proper constrained ("batch-constrained Q_{tot}") optimization, this is not the case for the the heuristic ("random Q_{tot}"), demonstrating the value of both joint batch optimization with constraints and LogEI.

generation is not an option since it is not able to incorporate the budget constraint. Therefore, joint batch optimization is required.

Here we give one such example in the context of Bayesian Optimization for sequential experimental design. We consider the five-dimensional silver nanoparticle flow synthesis problem from Liang et al. [39]. In this problem, to goal is to optimize the absorbance spectrum score of the synthesized nanoparticles over five parameters: four flow rate ratios of different components (silver, silver nitrate, trisodium citrate, polyvinyl alcohol) and a total flow rate Q_{tot}.

The original problem was optimized over a discrete set of parameterizations. For our purposes we created a continuous surrogate model based on the experimental dataset (available from `https://github.com/PV-Lab/Benchmarking`) by fitting an RBF interpolator (smoothing factor of 0.01) in `scipy` on the (negative) loss. We use the same search space as Liang et al. [39], but in addition to the box bounds on the parameters we also impose an additional constraint on the total flow rate $Q_{tot}^{max} = 2000$ µL/min across the batch: $\sum_{i=1}^{q} Q_{tot}^{i} \leq Q_{tot}^{max}$ (the maximum flow rate per syringe / batch element is 1000µL/min). This constraint expresses the maximum throughput limit of the microfluidic experimentation setup. The result of this constraint is that we cannot consider the batch elements (in this case automated syringe pumps) have all elements of a batch of experiments operate in the high-flow regime at the same time.

In our experiment, we use a batch size of $q = 3$ and start the optimization from 5 randomly sampled points from the domain. We run 75 replicates with random initial conditions (shared across the different methods), error bars show $\pm$ two times the standard error of the mean. Our baseline is uniform random sampling from the domain (we use a hit-and-run sampler to sample uniformly from the constraint polytope $\sum_{i=1}^{q} Q_{tot}^{i} \leq Q_{tot}^{max}$). We compare qEI vs. qLogEI, and for each of the two we evaluate (i) the version with the batch constraint imposed explicitly in the optimizer (the optimization in this case uses `scipy`'s `SLSQP` solver), and (ii) a heuristic that first samples the total flow rates $\{Q_{tot}^{i}\}_{i=1}^{q}$ uniformly from the constraint set, and then optimizes the acquisition function with the flow rates fixed to the sampled values.

The results in Figure 14 show that while both the heuristic ("random Q_{tot}") and the proper constrained optimization ("batch-constrained Q_{tot}") substantially outperform the purely random baseline, it requires uisng both LogEI *and* proper constraints to achieve additional performance gains over the other 3 combinations. Importantly, this approach is only possible by performing joint optimization of the batch, which underlines the importance of qLogEI and its siblings being able to achieve superior joint batch optimization in settings like this.

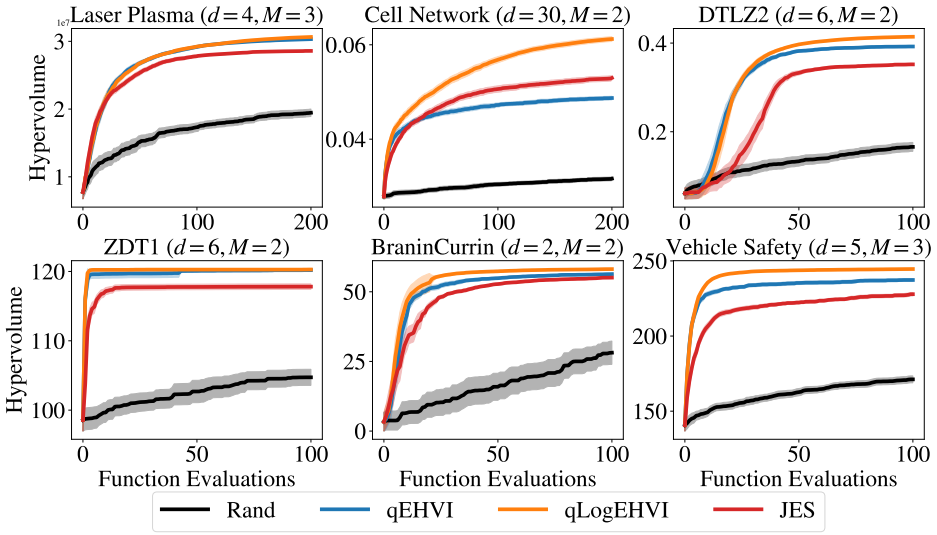

Figure 15: Sequential ($q = 1$) optimization performance on multi-objective problems, as measured by the hypervolume of the Pareto frontier across observed points. This plot includes JES [55]. Similar to the single-objective case, qLogEHVI significantly outperforms all baselines on all test problems.

D.5 Details on Multi-Objective Problems

We consider a variety of multi-objective benchmark problems. We evaluate performance on three synthetic biobjective problems Branin-Currin ($d = 2$) [4], ZDT1 ($d = 6$) [66], and DTLZ2 ($d = 6$) [10]. As described in 5, we also evaluated performance on three real world inspired problems. For the laser plasma acceleration problem, we used the public data available at Irshad et al. [27] to fit an independent GP surrogate model to each objective. We only queried te surrogate at the highest fidelity to create a single fidelity benchmark.

D.6 Effect of Temperature Parameter

In Figure 16, we examine the effect of fixed τ for the softplus operator on optimization performance. We find that smaller values typically work better.

D.7 Effect of the initialization strategy

Packages and frameworks commonly utilize smart initialization heuristics to improve acquisition function optimization performance. In Figure 17, we compare simple random restart optimization, where initial points are selected uniformly at random, with BoTorch's default initialization strategy, which evaluates the acquisition function on a large number of points selected from a scrambled Sobol sequence, and selects n points at random via Boltzman sampling (e.g., sampling using probabilities computed by taking a softmax over the acquisition values [2]. Here we consider 1024 initial candidates. We find that the BoTorch initialization strategy improves regret for all cases, and that qLogEI, followed by UCB show less sensitivity to the choice of initializations strategy. Figure18 examines the sensitivity of qEI to the number of initial starting points when performing standard random restart optimization and jointly optimizing the q points in the batch. We find that, consistent with our empirical and theoretical results in the main text, qEI often gets stuck in local minima for the Ackley test function, and additional random restarts often improve results but do not compensate for the fundamental optimality gap. The performance of qLogEI also improves as the number of starting points increases.

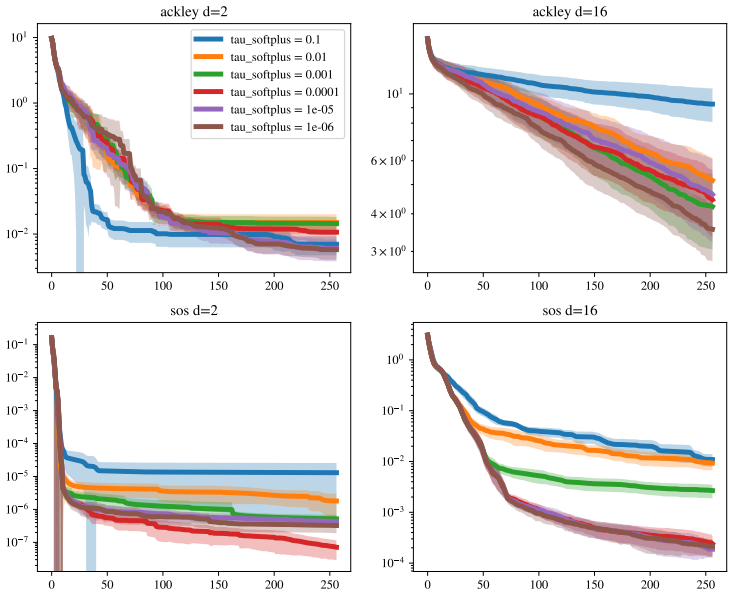

Figure 16: Ablation study on the convergence characteristics of LogEI on Ackley and sum of squares (SOS) problems in 2 and 16 dimensions. The study shows that it is important to choose a small τ_0 for the best convergence properties, which results in a very tight approximation to the original `ReLU` non-linearity in the integrand. Critically, setting τ_0 as low as 10^{-6} is only possible due to the transformation of all computations into log-space. Otherwise, the smoothed acquisition utility would exhibit similarly numerically vanishing gradients as the original `ReLU` non-linearity.

	Cell Network	Branin-Currin	DTLZ2	Laser Plasma	ZDT1	Vehicle Safety
JES	21.6 (+/- 1.1)	89.6 (+/- 3.3)	33.6 (+/- 1.0)	57.3 (+/- 0.7)	72.7 (+/- 1.0)	47.0 (+/- 1.6)
qEHVI	0.6 (+/- 0.0)	0.7 (+/- 0.0)	1.0 (+/- 0.0)	3.0 (+/- 0.1)	0.6 (+/- 0.0)	0.6 (+/- 0.0)
qLogEHVI	9.2 (+/- 0.8)	10.0 (+/- 0.4)	5.8 (+/- 0.2)	31.6 (+/- 1.7)	7.2 (+/- 0.7)	2.1 (+/- 0.1)
Rand	0.2 (+/- 0.0)	0.2 (+/- 0.0)	0.2 (+/- 0.0)	0.3 (+/- 0.0)	0.3 (+/- 0.0)	0.3 (+/- 0.0)

Table 1: Acquisition function optimization wall time in seconds on CPU (2x Intel Xeon E5-2680 v4 @ 2.40GHz) . We report the mean and $\pm$ 2 standard errors.

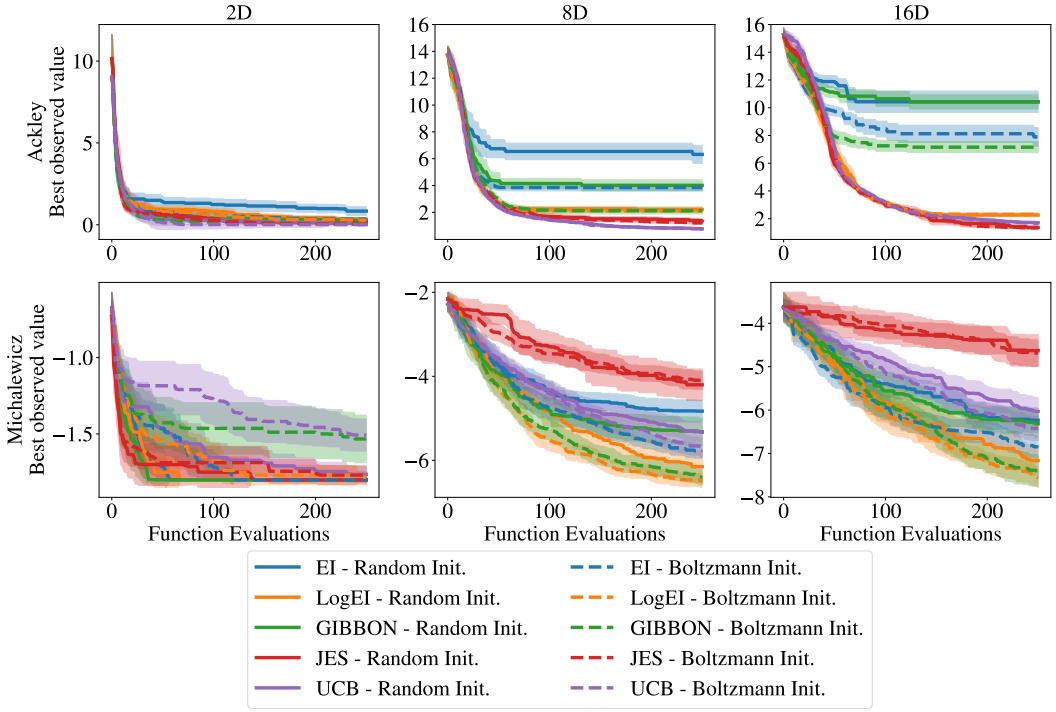

Figure 17: Sensitivity to the initialization strategy. Random selects random restart points from the design space uniformly at random, whereas Boltzmann initialization is the default BoTorch initialization strategy which selects points with higher acquisition function values with a higher probability via Boltzmann sampling.

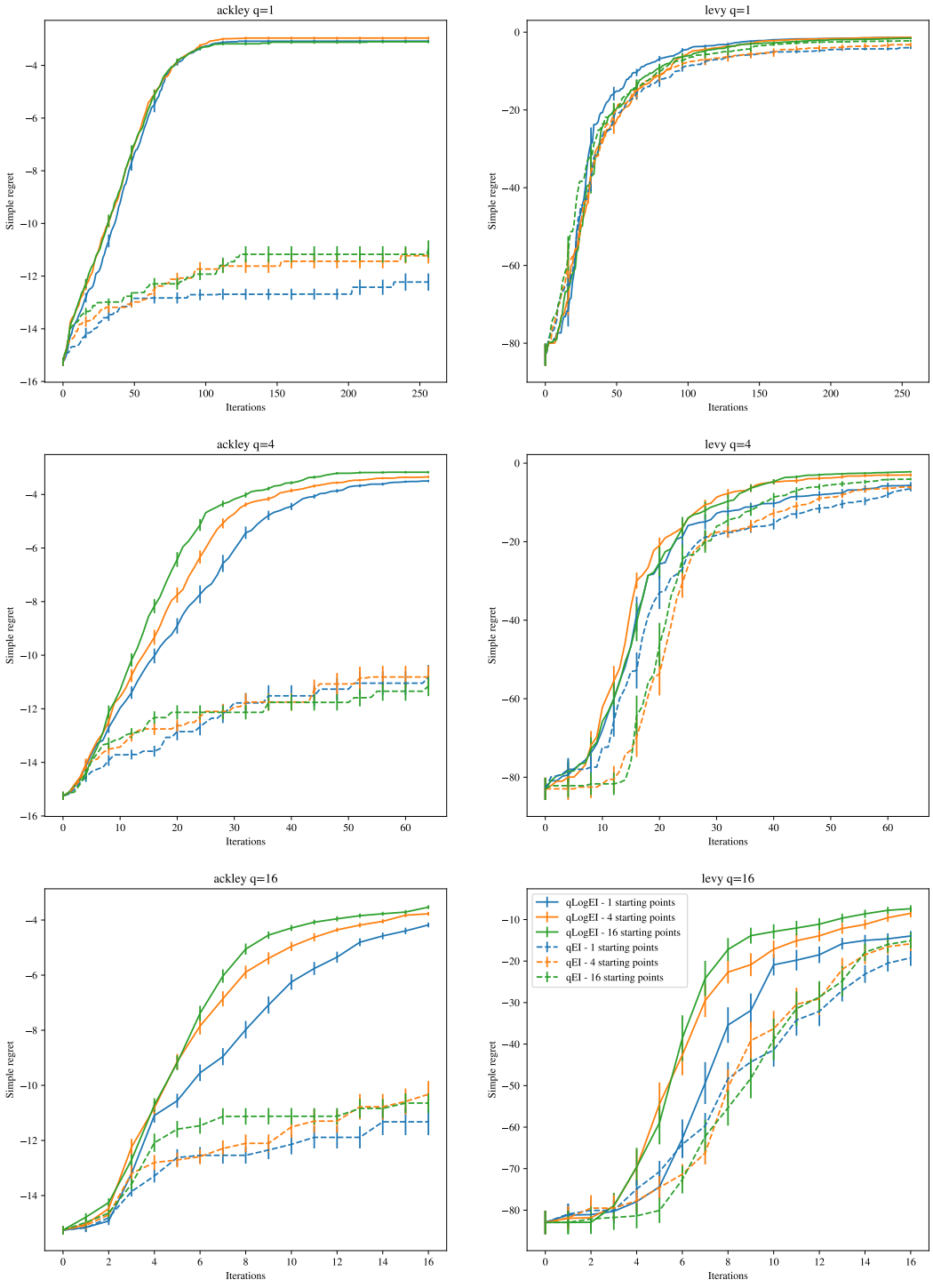

Figure 18: Sensitivity to number of starting points with multi-start optimization for the 16D Ackley and Levy test problems. Note: We plot negative regret, so higher is better.