Batched Energy-Entropy acquisition for Bayesian optimization

Felix Teufel [1 2] **Carsten Stahlhut** [1] **Jesper Ferkinghoff-Borg** [1]

Abstract

Bayesian optimization (BO) is a machine learning framework for performing sample-efficient global optimization of black-box problems. The optimization process is guided by an acquisition function that selects points to acquire in each round of BO. For life science problems, it is often most efficient to acquire points in parallel by conducting experiments in plate-based formats. However, commonly used acquisition functions are often high-dimensional and intractable in batch mode, leading to the use of sampling-based alternatives. We propose a statistical physics inspired acquisition function that can natively handle batches. Batched Energy-Entropy acquisition for BO (BEEBO) enables tight control of the explore-exploit trade-off of the optimization process. We demonstrate the applicability of BEEBO on a range of problems, showing competitive performance to existing acquisition functions.

1. Introduction

Bayesian Optimization (BO) has since its inception (Kushner, 1964; Močkus, 1975) made a profound contribution to the realm of global optimization of black-box functions through the usage of Bayesian statistics. For global optimization problems pursuing $x_* = \text{argmax}_{x \in \mathcal{X}} f_{\text{true}}(x)$, BO has surfaced as a premier strategy for efficiently handling especially complex and costly unknown functions. While BO is traditionally formulated in a single-point scenario, where individual points are queried and results are observed sequentially, there are situations where *batched* acquisition is needed. Such situations arise when $f_{\text{true}}(x)$ is expensive to evaluate in either time or cost, but can be effectively evaluated in parallel by dispatching multiple experiments, reducing the overall optimization time. This is often the case in e.g. drug discovery, materials design or hyperparameter tuning for deep models (Snoek et al., 2012; Dodge et al., 2017; Griffiths & Hernández-Lobato, 2020; Park et al., 2022; Stanton et al., 2022).

Default BO has been generalized to accommodate noisy inputs (Frazier, 2018; Daulton et al., 2022), heteroskedastic noise (Letham et al., 2019; Makarova et al., 2021), application to multi-task problems (Swersky et al., 2013), multi-fidelity (Takeno et al., 2020), high-dimensional input spaces (Moriconi et al., 2019), and parallel methods with batch queries (Azimi et al., 2010; Kathuria et al., 2016). Generally, these properties are addressed by customizing one of the two key components in BO, either the *surrogate model* or the *acquisition function*. The surrogate model f approximates the black-box function f_{true} using the available data. Typically, the model of choice is a Gaussian Process (GP) (Williams & Rasmussen, 1995). The acquisition function is responsible for guiding the selection of new input point(s) to evaluate at each optimization step, utilizing f to identify promising regions in the input domain and exploring the

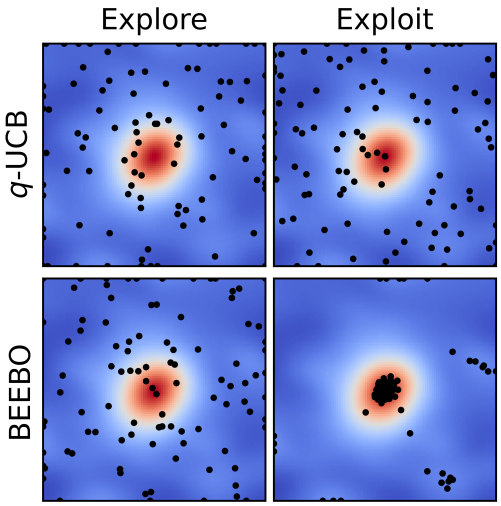

Figure 1. q-UCB does not allow for controlling its explore-exploit trade-off with large batches. A GP surrogate (background) was initialized with 100 random points of the Ackley function. q-UCB was run with $\beta = 0.1$ and $\beta = 100$, BEEBO with $T' = 0.05$ and $T' = 50$. $q = 100$.

[1]Machine Intelligence, Novo Nordisk A/S [2]Department of Biology, University of Copenhagen. Correspondence to: Felix Teufel <fegt@novonordisk.com>, Jesper Ferkinghoff-Borg <jfgb@novonordisk.com>.

Accepted at the 1st Machine Learning for Life and Material Sciences Workshop at ICML 2024.

unknown function further.

Any acquisition process needs to trade off exploration (reducing uncertainty to learn a better surrogate model) against exploitation (selecting points with a high expected $f_{\text{true}}(x)$ based on the current surrogate). In this work, we are particularly interested in acquisition processes that make this trade-off controllable using a hyperparameter. Controllability can be a desirable property if e.g. domain knowledge relating to the difficulty of the optimization process and the quality of the surrogate model is available, or if the strategy needs to be adjusted depending on future experimental budgets and the stage of the discovery process. It can also be desirable to acquire multiple x with high $f_{\text{true}}(x)$ in a batch (as opposed to just finding the optimum x_*, with the remaining x being considered explorative). This is useful when optima identified in BO can be subject to constraints that are unknown at optimization time, but may render x_* intractable (Maus et al., 2023). Practical examples include e.g. the synthesizability of a material at larger scale, when BO experiments are performed at lab scale; or the *in vivo* activity of a molecule with BO experiments performed *in vitro*.

A wide range of batch-mode acquisition functions has been proposed, with approaches often leveraging random sampling strategies or Monte Carlo (MC) integration, which can adversely affect controllability (Figure 1). In contrast, we here introduce BEEBO (Batched Energy-Entropy acquisition for BO), a statistical physics inspired acquisition function that natively generalizes to batched acquisition. BEEBO enables

- Parallel gradient-based optimization of the inputs, **without requiring approximations, sampling or Monte Carlo integrals**.
- **Tight control of the explore-exploit trade-off** using a single temperature hyperparameter.
- Risk-averse BO under **heteroskedastic noise**.

We demonstrate the application of BEEBO on a wide range of test problems, and investigate its behaviour under heteroskedastic noise.

2. Related works

Batch variants of traditional strategies Parallel acquisition approaches in BO often start from established single-point acquisition functions like probability of improvement (PI), expected improvement (EI), knowledge gradient (KG) ,and upper confidence bound (UCB) (Močkus, 1975; Jones et al., 1998; Auer, 2002; Frazier et al., 2009; Srinivas et al., 2009). Reformulating these to batch mode with Q query points, we obtain q-PI, q-EI and q-UCB (Wilson et al., 2017; 2018). The resulting batch expressions are usually challenging to optimize, typically involving greedy algorithms (Shah & Ghahramani, 2015) or an integral expression over multiple points without a closed-form solution, requiring MC integration. Of particular interest is Wilson et al. (2018), in which they adopt the reparameterization trick (Kingma et al., 2015; Rezende et al., 2014).

Greedy strategies Batch filling strategies that score candidate points sequentially can be used with single-point acquisition functions. Kriging Believer (KB) (Ginsbourger et al., 2010) uses EI to select points and iteratively updates the GP by fantasizing an observation with the posterior mean. Likewise, GP-BUCB (Desautels et al., 2014) uses fantasized observations to update $\sqrt{C(x)}$ at each step. Gonzalez et al. (2016) introduce local penalization (LP) that repulses selection away from already selected points.

Entropy based strategies From an information theory perspective, BO can be interpreted as seeking to reduce uncertainty over the optima of the unknown function. General-purpose Information-Based Bayesian OptimizatioN (GIBBON) (Moss et al., 2021) proposes a lower bound formulation for the intractable batch criterion of max-value entropy search (Wang & Jegelka, 2017; Takeno et al., 2020). Despite being formulated to handle a large degree of parallelism, Moss et al. (2021) reported that GIBBON fails in practice for large batches with $Q > 50$, potentially, as a consequence of the accuracy of the lower bound approximation.

Thompson sampling Given the challenges of generalizing acquisition functions to batch mode, Thompson sampling (TS) (Thompson, 1933; Chowdhury & Gopalan, 2017; Kandasamy et al., 2018; Mirrokni et al., 2021; Karbasi et al., 2021) is a popular alternative strategy for guiding batched BO. While being an attractive approach, it has been demonstrated that default TS can become too exploitative (Adachi et al., 2023), motivating the use of advanced strategies on top of TS that ensure diversity (Maus et al., 2023).

3. The BEEBO acquisition function

Assume $f_{\text{true}} : X \to \mathbb{R}$ is some real output associated with the input and a set of data be given $D = \{(x_i, y_i)\}_{i=1}^{N}$ where $y_i \in \mathbb{R}$ represent some noisy observations of $f_{\text{true}}(x_i)$, say

$$y_i = f_{\text{true}}(x_i) + \epsilon_i \tag{1}$$

with ϵ_i denoting the measurement noise. Let $\mathbf{x} = (x_1, \cdots, x_Q) \in X^Q$ represent a collection of test points we wish to assign an acquisition value to. In keeping with the BO framework, we assume a given posterior probability distribution over the surrogate function $\mathbf{f}$ evaluated at $\mathbf{x}$,

$$\mathbf{f}(\mathbf{x}) \sim P(\mathbf{f} \mid D, \mathbf{x}) \tag{2}$$

The lack of knowledge we have of the surrogate function at $\mathbf{x}$ is quantified by the differential *entropy* H:

$$H\big(\mathbf{f} \mid D, \mathbf{x}\big) = -\int P\big(\mathbf{f} \mid D, \mathbf{x}\big) \ln\Big(P\big(\mathbf{f} \mid D, \mathbf{x}\big)\Big) d\mathbf{f} \quad (3)$$

This entropy can be contrasted with the expected entropy of the surrogate function if Q observations $\mathbf{y} = (y_1, \cdots, y_Q)$ were acquired at $\mathbf{x}$, ie. if the training data D would be augmented with $D'(\mathbf{y}) = \{(x_q, y_q)\}_{q=1}^{Q}$, to form the joint data set $D_{\text{aug}}(\mathbf{y}) = \big(D, D'(\mathbf{y})\big)$. We refer to this entropy as H_{aug}:

$$H_{\text{aug}}\big(\mathbf{f} \mid D, \mathbf{x}\big) = \int P(\mathbf{y} \mid D, \mathbf{x}) H\big(\mathbf{f} \mid D_{\text{aug}}(\mathbf{y})\big) d\mathbf{y}, \quad (4)$$

where $P(\mathbf{y} \mid D, \mathbf{x})$ represent the posterior predictive distribution at $\mathbf{x}$. The expected *information gain*, $I(\mathbf{x})$, from acquiring observations at $\mathbf{x}$ is given by the expected reduction of entropy from this process:

$$I(\mathbf{x}) = H\big(\mathbf{f} \mid D, \mathbf{x}\big) - H_{\text{aug}}\big(\mathbf{f} \mid D, \mathbf{x}\big) \quad (5)$$

We propose to represent the explore component of the acquisition function, a_{BEEBO}, by $I(\mathbf{x})$. The information gain $I(\mathbf{x})$ is distinct from the quantities exploited by entropy search approaches, as it quantifies global uncertainty reduction, rather than estimating the information over an unknown x_*. The information gain is directly applicable to multivariate functions and to heteroskedastic settings where $\sigma^2 = \sigma^2(\mathbf{x})$. Since large measurement uncertainties imply smaller information gain, a_{BEEBO} exhibits risk-averse behaviour (Makarova et al., 2021) by prioritizing regions of small uncertainties from where more precise information of f_{true} can be obtained, everything else being equal.

The exploit component of BEEBO relies on taking expectation values of a scalar function of the random variable $\mathbf{f}(\mathbf{x})$, $\tilde{E} : \mathbb{R}^Q \to \mathbb{R}$, that summarizes the optimality properties of a given batch $\mathbf{x}$. Natural choices would be the mean or the maximum of $\mathbf{f}(\mathbf{x})$. Of particular interest is expressing the optimality as a softmax-weighted sum over $\mathbf{f}(\mathbf{x})$, as this allows us to smoothly interpolate between the two regimes.

$$E(\mathbf{x}) = -\mathbb{E}[\tilde{E}(\mathbf{x})] \cdot Q = -\mathbb{E}\left[\sum_{q=1}^{Q} \text{softmax}(\beta \mathbf{f})_q f_q\right] \cdot Q, \quad (6)$$

where β is the softmax inverse temperature. At $\beta = 0$, we recover the mean (we refer to this limit as *meanBEEBO*). We scale the expectation with Q so that both I and E scale linearly. While the mean provides a closed form expression for its expectation, this is not the case for the general softmax-weighted sum of a multivariate normal. Using Taylor expansion, we introduce an approximation of the expectation of the softmax-weighted sum that is fully differentiable and can be computed in closed form (*maxBEEBO*). A detailed derivation is provided in the Appendix.

The BEEBO acquisition function then takes the form

$$a_{\text{BEEBO}}(\mathbf{x}) = -E(\mathbf{x}) + T \cdot I(\mathbf{x}), \quad (7)$$

where T sets the balance between exploitation (small T) and exploration (large T). This acquisition function bears a strong similarity to the definition of (negative) *free energies* in statistical physics, where E and I correspond to respectively the thermodynamic energy and entropy of the system and T corresponds to the temperature.

Closed-form expressions for the inference step are available when using a GP surrogate model, with the GP posterior being a multivariate Gaussian distribution $\mathcal{N}(\cdot \mid \mu, C)$ with mean μ, and covariance C. In a GP, C only depends on the input location of the test points $\mathbf{x}$ and the data points $\mathbf{x}_D$ with their corresponding measurement uncertainties, so we can also compute H_{aug} in closed form. All operations needed to compute the acquisition value $a_{\text{BEEBO}}(\mathbf{x})$ are thus analytical, and a batch of points $\mathbf{x}$ can be optimized with gradient-based methods using automatic differentiation.

4. Experiments

We benchmark acquisition function performance on a range of maximization test problems with varying dimensions available in BoTorch (Balandat et al., 2020). On each test problem, we perform 10 rounds of BO using q-UCB or BEEBO with a given explore-exploit parameter for direct comparison. We configure BEEBO so that its parameter is directly comparable to q-UCB. To mimic the real-world use case of plate-based parallel assays, we perform experiments with a batch size of $Q = 100$. We run BO for 10 rounds, with the last round running in full-exploit mode.

We report the mean best observed objective value after 10 rounds over the five replicates. Results are min-max normalized using the best starting value and the true optimum $f_{\text{true}}(x^*)$ of the problem. As we are not only interested in identifying a single x with good $f_{\text{true}}(x)$, we additionally quantify the overall quality of the final (exploitative) batch using the batch instantaneous regret $R = \sum_{q<Q} f_{\text{true}}(x^*) - f_{\text{true}}$. We normalize this metric by dividing it with the R of a random batch of Q points, yielding R_{rel}.

5. Results

We benchmark BEEBO against q-UCB at three rates of κ. Overall, we find that the simpler meanBEEBO variant outperforms maxBEEBO in terms of mean performance on all but the lowest rate of κ (Table 1). As we consider the

Table 1. Highest observed value after 10 rounds of BO with $Q = 100$. The best value at each κ is indicated in blue. BEEBO is configured with $T' = 1/2\sqrt{\kappa}$. Full BO curves are provided in the Appendix.

Problem	d	$\sqrt{\kappa} = 0.1$			$\sqrt{\kappa} = 1.0$			$\sqrt{\kappa} = 10.0$		
		meanBEEBO	maxBEEBO	q-UCB	meanBEEBO	maxBEEBO	q-UCB	meanBEEBO	maxBEEBO	q-UCB
Ackley	2	0.993	0.976	0.961	0.975	0.965	0.973	0.972	0.982	0.996
Levy	2	1	1	1	1	1	0.999	1	0.998	0.997
Rastrigin	2	0.977	0.996	0.938	0.988	0.983	0.982	0.974	0.991	0.935
Rosenbrock	2	0.992	0.998	0.949	0.962	0.975	0.905	0.980	0.975	0.984
Styblinski-Tang	2	0.962	1	1	1	1	1	1	1	0.999
Shekel	4	0.509	0.327	0.284	0.902	0.336	0.391	0.755	0.418	0.266
Hartmann	6	1	0.896	0.91	1	0.974	0.957	0.978	0.967	0.895
Cosine	8	1	0.999	0.940	0.999	0.975	0.926	0.644	0.929	0.716
Ackley	10	0.913	0.829	0.807	0.888	0.726	0.775	0.828	0.543	0.520
Levy	10	0.989	0.964	0.885	0.964	0.945	0.898	0.953	0.911	0.606
Powell	10	0.987	0.952	0.908	0.966	0.954	0.909	0.899	0.931	0.256
Rastrigin	10	0.483	0.532	0.413	0.503	0.512	0.537	0.598	0.606	0.299
Rosenbrock	10	0.993	0.992	0.965	0.993	0.987	0.966	0.929	0.971	0.710
Styblinski-Tang	10	0.797	0.784	0.233	0.812	0.643	0.368	0.242	0.217	0.072
Ackley	20	0.843	0.819	0.744	0.857	0.788	0.753	0.789	0.390	0.566
Levy	20	0.936	0.953	0.92	0.939	0.901	0.899	0.896	0.911	0.786
Powell	20	0.947	0.936	0.951	0.966	0.880	0.905	0.840	0.908	0.819
Rastrigin	20	0.373	0.514	0.431	0.462	0.480	0.503	0.518	0.491	0.401
Rosenbrock	20	0.993	0.991	0.973	0.993	0.982	0.979	0.906	0.923	0.911
Styblinski-Tang	20	0.706	0.607	0.187	0.669	0.417	0.542	0.305	0.279	0.029
Ackley	50	0.221	0.622	0.622	0.146	0.705	0.758	0.842	0.457	0.732
Levy	50	0.976	0.977	0.951	0.978	0.955	0.955	0.943	0.867	0.933
Powell	50	0.940	0.976	0.970	0.978	0.970	0.953	0.959	0.929	0.978
Rastrigin	50	0.273	0.473	0.446	0.505	0.466	0.405	0.453	0.445	0.501
Rosenbrock	50	0.979	0.978	0.972	0.984	0.981	0.987	0.988	0.968	0.987
Styblinski-Tang	50	0.605	0.536	0.341	0.693	0.415	0.713	0.417	0.332	0.416
Ackley	100	0.310	0.526	0.711	0.347	0.536	0.634	0.864	0.707	0.848
Emb. Hartmann 6	100	0.980	0.982	0.894	0.988	0.933	0.923	0.916	0.914	0.921
Levy	100	0.890	0.962	0.954	0.966	0.942	0.938	0.943	0.946	0.965
Powell	100	0.795	0.985	0.982	0.946	0.981	0.978	0.987	0.981	0.984
Rastrigin	100	0.522	0.481	0.437	0.367	0.479	0.453	0.467	0.469	0.443
Rosenbrock	100	0.812	0.928	0.97	0.975	0.978	0.966	0.975	0.975	0.983
Styblinski-Tang	100	0.564	0.432	0.307	0.470	0.331	0.574	0.396	0.309	0.289
Mean		0.796	0.816	0.756	0.824	0.791	0.800	0.793	0.747	0.689
Median		0.936	0.952	0.910	0.964	0.942	0.905	0.899	0.911	0.786

configuration with the lowest rate to be exploit-dominated, this can be understood as a consequence of maxBEEBO effectively releasing non-contributing points for further exploration, with the low explore rate seeming sufficient to induce the necessary diversity.

While results on individual test problems vary, meanBEEBO shows improved performance over q-UCB especially in the medium dimension range up to 50. For d=100, we find mixed performance, with meanBEEBO gradually becoming more competitive with increasing κ. This is due to the fact that with increasing dimensionality, more exploration is beneficial for learning a good surrogate model before an actual BO process becomes effective. As we find that q-UCB inherently performs more random-like sampling at large Q, irrespective of κ, it benefits in such situations.

On average over all 33 performed experiments, meanBEEBO improves upon q-UCB at any of the three rates. We additionally benchmarked BEEBO against other popular BO strategies without explore-exploit hyperparameters. Interestingly, we found that the Kriging Believer (KB) iterative heuristic (Ginsbourger et al., 2010) can perform very competitively for large batches when using LogEI (Ament et al., 2023) as the acquisition function (Table A1).

When evaluating R_{rel} in the ultimate round of BO, we find that BEEBO allows us to effectively acquire a batch with high $f_{\text{true}}(x)$, highlighting the controllability of the acquisition function (Table 2). The R_{rel} of q-UCB is only slightly better than the R of a random batch in many cases, even though the explore component was explicitly set to 0. We note that this is not due to the surrogate function being unsuitable - the results in Table 1 indicate that in most cases the location of $f_{\text{true}}(x^*)$ is approximately known by round 10. Rather, we assume that this a consequence of the challenges of MC-based optimization of the acquisition function at large Q.

6. Discussion

We introduce BEEBO, an acquisition function for BO with GPs that can be optimized analytically and scales natively to batched acquisition. By exploiting the independence of the information gain $I(\mathbf{x})$ on measurements $\mathbf{y}$ when using

Table 2. Relative batch instantaneous regret R_{rel} in round 10 ($\kappa = 0$) with $Q = 100$. The best value at each κ is indicated in blue. BEEBO is configured with $T' = 1/2\sqrt{\kappa}$. Lower means better.

PROBLEM	D	$\sqrt{\kappa} = 0.1$			$\sqrt{\kappa} = 1.0$			$\sqrt{\kappa} = 10.0$		
		MEANBEEBO	MAXBEEBO	q-UCB	MEANBEEBO	MAXBEEBO	q-UCB	MEANBEEBO	MAXBEEBO	q-UCB
ACKLEY	2	0.28	0.25	1.01	0.23	0.24	0.99	0.25	0.14	1.01
LEVY	2	0.13	0.12	1.38	0.09	0.09	0.99	0.09	0.11	1.16
RASTRIGIN	2	0.43	0.66	1.03	0.44	0.48	1.00	0.51	0.53	1.04
ROSENBROCK	2	0.00	0.00	0.86	0.00	0.00	0.92	0.00	0.00	0.87
STYBLINSKI-TANG	2	0.17	0.17	1.01	0.17	0.17	1.02	0.17	0.17	1.06
SHEKEL	4	0.84	0.75	0.99	0.66	0.72	0.99	0.68	0.67	0.99
HARTMANN	6	0.05	0.22	0.93	0.08	0.10	0.97	0.10	0.11	0.86
COSINE	8	0.00	0.00	0.95	0.00	0.02	0.96	0.23	0.05	0.90
ACKLEY	10	0.42	0.34	0.93	0.32	0.32	0.94	0.24	0.46	0.94
LEVY	10	0.04	0.03	1.26	0.02	0.04	0.97	0.23	0.10	1.14
POWELL	10	0.01	0.03	1.01	0.01	0.07	1.06	0.06	0.14	1.23
RASTRIGIN	10	0.64	0.46	0.91	0.48	0.51	0.89	0.59	0.45	0.90
ROSENBROCK	10	0.00	0.01	0.91	0.00	0.01	0.77	0.07	0.05	0.93
STYBLINSKI-TANG	10	0.21	0.23	1.26	0.23	0.33	1.12	0.58	0.49	1.23
ACKLEY	20	0.67	0.22	0.95	0.26	0.31	0.95	0.21	0.61	0.91
LEVY	20	0.08	0.18	0.86	0.08	0.11	0.90	0.12	0.21	1.04
POWELL	20	0.10	0.08	0.87	0.01	0.08	0.70	0.03	0.12	0.92
RASTRIGIN	20	0.71	0.62	0.86	0.61	0.64	0.85	0.51	0.56	0.85
ROSENBROCK	20	0.05	0.12	0.59	0.00	0.05	0.61	0.04	0.05	0.94
STYBLINSKI-TANG	20	0.40	0.40	1.10	0.36	0.53	1.14	0.73	0.56	1.16
ACKLEY	50	0.90	0.40	0.94	0.86	0.47	0.94	0.16	0.54	0.87
LEVY	50	0.04	0.02	0.61	0.03	0.05	0.69	0.04	0.24	0.95
POWELL	50	0.02	0.02	0.42	0.02	0.04	0.54	0.01	0.08	0.81
RASTRIGIN	50	0.75	0.59	0.80	0.60	0.59	0.79	0.89	0.58	0.76
ROSENBROCK	50	0.02	0.01	0.46	0.01	0.04	0.47	0.01	0.05	0.61
STYBLINSKI-TANG	50	0.46	0.45	0.97	1.06	0.57	1.18	0.71	0.72	0.99
ACKLEY	100	0.68	0.55	0.95	0.82	0.48	0.94	0.14	0.29	0.87
EMB. HARTMANN 6	100	0.09	0.14	0.57	0.04	0.10	0.88	0.18	0.19	0.72
LEVY	100	0.09	0.04	0.63	0.04	0.17	0.72	0.04	0.06	0.60
POWELL	100	0.11	0.01	0.43	0.03	0.03	0.54	0.01	0.01	0.58
RASTRIGIN	100	0.50	0.47	0.76	0.58	0.55	0.83	0.55	0.55	0.77
ROSENBROCK	100	0.12	0.06	0.53	0.02	0.06	0.62	0.02	0.04	0.54
STYBLINSKI-TANG	100	0.37	0.55	0.91	0.45	0.63	1.20	0.51	0.77	0.95
MEAN		0.28	0.25	0.87	0.26	0.26	0.88	0.26	0.29	0.91
MEDIAN		0.13	0.18	0.91	0.09	0.17	0.94	0.17	0.19	0.92

GP surrogates, BEEBO models the interdependence of unknown points $\mathbf{x}$ in a batch and can optimize their positions jointly using gradient descent. BEEBO enables full control of its explore-exploit trade-off using a hyperparameter T that directly balances two terms, akin to UCB. Unlike in the reparametrization-based q-methods, BEEBO's T has predictable behaviour also at increasing batch sizes.

7. Outlook

In our experiments, we have focused on maintaining consistent explore-exploit ratios throughout the optimization rounds to ensure an equitable experimental comparison with q-UCB and demonstrate the effect of the hyperparameter choice. However, a more dynamic approach involving variable ratios could be more effective in real-world applications with a predetermined number of rounds (De Ath et al., 2021).

In this work, following established practices in BO research, we leveraged test problems with known structure to perform comprehensive benchmarking of acquisition strategies without incurring the prohibitive cost of real-world experiments. In the future, it will be interesting to use BEEBO in BO frameworks for molecular design. As BEEBO does not rely on any assumptions beyond the surrogate being a GP, it could be used as a drop-in replacement for any other strategy.

We note that the BEEBO expression could naturally be extended to multi-objective optimization problems by capitalizing on GPs that handle vector-valued functions, such as multi-task GPs (Bonilla et al., 2007; Swersky et al., 2013). Through e.g. the usage of the *intrinsic model of coregionalization*, we obtain a covariance function k, and thereby a covariance matrix $C(\mathbf{x})$, over all input-task pairs. As the multi-task covariance matrix is jointly Gaussian, the expression of the information gain remains unchanged and can be computed like in the single-task case. The energy $E(\mathbf{x})$ becomes vector-valued, providing an energy term for each of the tasks. This would allow for the introduction of task-specific weights in the acquisition function.

Availability

A BoTorch implementation of BEEBO is available at

https://github.com/fteufel/BEE-BO.

References

Adachi, M., Hayakawa, S., Hamid, S., Jørgensen, M., Oberhauser, H., and Osborne, M. A. Sober: Scalable batch bayesian optimization and quadrature using recombination constraints. *arXiv preprint arXiv:2301.11832*, 2023.

Ament, S., Daulton, S., Eriksson, D., Balandat, M., and Bakshy, E. Unexpected improvements to expected improvement for bayesian optimization. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=1vyAG6j9PE.

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

Azimi, J., Fern, A., and Fern, X. Batch bayesian optimization via simulation matching. *Advances in Neural Information Processing Systems*, 23, 2010.

Balandat, M., Karrer, B., Jiang, D. R., Daulton, S., Letham, B., Wilson, A. G., and Bakshy, E. Botorch: a framework for efficient monte-carlo bayesian optimization. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Bonilla, E. V., Chai, K., and Williams, C. Multi-task gaussian process prediction. In Platt, J., Koller, D., Singer, Y., and Roweis, S. (eds.), *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2007. URL https://proceedings.neurips.cc/paper_files/paper/2007/file/66368270ffd51418ec58bd793f2d9b1b-Paper.pdf.

Charlier, B., Feydy, J., Glaunès, J. A., Collin, F.-D., and Durif, G. Kernel operations on the gpu, with autodiff, without memory overflows. *Journal of Machine Learning Research*, 22(74):1–6, 2021. URL http://jmlr.org/papers/v22/20-275.html.

Chowdhury, S. R. and Gopalan, A. On kernelized multi-armed bandits. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 844–853. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/chowdhury17a.html.

Daulton, S., Cakmak, S., Balandat, M., Osborne, M. A., Zhou, E., and Bakshy, E. Robust Multi-Objective Bayesian Optimization Under Input Noise. *arXiv*, 2022. doi: 10.48550/arxiv.2202.07549.

De Ath, G., Everson, R. M., Rahat, A. A. M., and Fieldsend, J. E. Greed is good: Exploration and exploitation trade-offs in bayesian optimisation. *ACM Trans. Evol. Learn. Optim.*, 1(1), apr 2021. doi: 10.1145/3425501. URL https://doi.org/10.1145/3425501.

Desautels, T., Krause, A., and Burdick, J. W. Parallelizing exploration-exploitation tradeoffs in gaussian process bandit optimization. *Journal of Machine Learning Research*, 15(119):4053–4103, 2014. URL http://jmlr.org/papers/v15/desautels14a.html.

Dodge, J., Jamieson, K., and Smith, N. A. Open loop hyperparameter optimization and determinantal point processes. *arXiv preprint arXiv:1706.01566*, 2017.

Eriksson, D. and Jankowiak, M. High-dimensional Bayesian optimization with sparse axis-aligned subspaces. In de Campos, C. and Maathuis, M. H. (eds.), *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pp. 493–503. PMLR, 27–30 Jul 2021. URL https://proceedings.mlr.press/v161/eriksson21a.html.

Frazier, P., Powell, W., and Dayanik, S. The knowledge-gradient policy for correlated normal beliefs. *INFORMS journal on Computing*, 21(4):599–613, 2009.

Frazier, P. I. A Tutorial on Bayesian Optimization. *arXiv*, 2018. doi: 10.48550/arxiv.1807.02811.

Gardner, J. R., Pleiss, G., Bindel, D., Weinberger, K. Q., and Wilson, A. G. Gpytorch: blackbox matrix-matrix gaussian process inference with gpu acceleration. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pp. 7587–7597, Red Hook, NY, USA, 2018. Curran Associates Inc.

Ginsbourger, D., Le Riche, R., and Carraro, L. *Kriging Is Well-Suited to Parallelize Optimization*, pp. 131–162. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010. ISBN 978-3-642-10701-6. doi: 10.1007/978-3-642-10701-6_6. URL https://doi.org/10.1007/978-3-642-10701-6_6.

Gonzalez, J., Dai, Z., Hennig, P., and Lawrence, N. Batch bayesian optimization via local penalization. In Gretton, A. and Robert, C. C. (eds.), *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pp. 648–657, Cadiz, Spain, 09–11 May

2016. PMLR. URL https://proceedings.mlr.press/v51/gonzalez16a.html.

Griffiths, R.-R. and Hernández-Lobato, J. M. Constrained bayesian optimization for automatic chemical design using variational autoencoders. *Chemical science*, 11(2): 577–586, 2020.

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

Kandasamy, K., Krishnamurthy, A., Schneider, J., and Póczos, B. Parallelised bayesian optimisation via thompson sampling. In *International Conference on Artificial Intelligence and Statistics*, pp. 133–142. PMLR, 2018.

Karbasi, A., Mirrokni, V., and Shadravan, M. Parallelizing thompson sampling. *Advances in Neural Information Processing Systems*, 34:10535–10548, 2021.

Kathuria, T., Deshpande, A., and Kohli, P. Batched gaussian process bandit optimization via determinantal point processes. *Advances in neural information processing systems*, 29, 2016.

Kingma, D. P., Salimans, T., and Welling, M. Variational dropout and the local reparameterization trick. *Advances in neural information processing systems*, 28, 2015.

Kushner, H. J. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. 1964.

Letham, B., Karrer, B., Ottoni, G., and Bakshy, E. Constrained bayesian optimization with noisy experiments. 2019.

Makarova, A., Usmanova, I., Bogunovic, I., and Krause, A. Risk-averse heteroscedastic bayesian optimization. *Advances in Neural Information Processing Systems*, 34: 17235–17245, 2021.

Maus, N., Wu, K., Eriksson, D., and Gardner, J. Discovering many diverse solutions with bayesian optimization. In Ruiz, F., Dy, J., and van de Meent, J.-W. (eds.), *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pp. 1779–1798. PMLR, 25–27 Apr 2023. URL https://proceedings.mlr.press/v206/maus23a.html.

Mirrokni, V., Shadravan, M., et al. Parallelizing thompson sampling. In *Advances in Neural Information Processing Systems*, 2021.

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

Moriconi, R., Deisenroth, M. P., and Kumar, K. S. S. High-dimensional Bayesian optimization using low-dimensional feature spaces. *arXiv*, 2019. doi: 10.48550/arxiv.1902.10675.

Moss, H. B., Leslie, D. S., González, J. I., and Rayson, P. Gibbon: General-purpose information-based bayesian optimisation. *ArXiv*, abs/2102.03324, 2021. URL https://api.semanticscholar.org/CorpusID:231839492.

Papenmeier, L., Nardi, L., and Poloczek, M. Increasing the scope as you learn: Adaptive bayesian optimization in nested subspaces. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=e4Wf6112DI.

Park, J. W., Stanton, S., Saremi, S., Watkins, A., Dwyer, H., Gligorijevic, V., Bonneau, R., Ra, S., and Cho, K. PropertyDAG: Multi-objective Bayesian optimization of partially ordered, mixed-variable properties for biological sequence design. *arXiv*, 2022. doi: 10.48550/arxiv.2210.04096.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pp. 1278–1286. PMLR, 2014.

Shah, A. and Ghahramani, Z. Parallel Predictive Entropy Search for Batch Global Optimization of Expensive Objective Functions. *Advances in neural information processing systems*, 28, 2015. doi: 10.48550/arxiv.1511.07130.

Snoek, J., Larochelle, H., and Adams, R. P. Practical Bayesian Optimization of Machine Learning Algorithms. *Advances in neural information processing systems*, 25, 2012.

Srinivas, N., Krause, A., Kakade, S. M., and Seeger, M. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.

Stanton, S., Maddox, W., Gruver, N., Maffettone, P., Delaney, E., Greenside, P., and Wilson, A. G. Accelerating Bayesian optimization for biological sequence design with denoising autoencoders. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S. (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 20459–20478. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/stanton22a.html.

Swersky, K., Snoek, J., and Adams, R. P. Multi-task bayesian optimization. *Advances in neural information processing systems*, 26, 2013.

Takeno, S., Fukuoka, H., Tsukada, Y., Koyama, T., Shiga, M., Takeuchi, I., and Karasuyama, M. Multi-fidelity bayesian optimization with max-value entropy search and its parallelization. In *International Conference on Machine Learning*, pp. 9334–9345. PMLR, 2020.

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

Wang, Z. and Jegelka, S. Max-value entropy search for efficient bayesian optimization. In *International Conference on Machine Learning*, pp. 3627–3635. PMLR, 2017.

Wang, Z., Hutter, F., Zoghi, M., Matheson, D., and De Freitas, N. Bayesian optimization in a billion dimensions via random embeddings. *J. Artif. Int. Res.*, 55(1):361–387, jan 2016. ISSN 1076-9757.

Williams, C. and Rasmussen, C. Gaussian processes for regression. *Advances in neural information processing systems*, 8, 1995.

Wilson, J. T., Moriconi, R., Hutter, F., and Deisenroth, M. P. The reparameterization trick for acquisition functions. *arXiv*, 2017. doi: 10.48550/arxiv.1712.00424.

Wilson, J. T., Hutter, F., and Deisenroth, M. P. Maximizing acquisition functions for Bayesian optimization. *arXiv*, 2018. doi: 10.48550/arxiv.1805.10196.

A. Appendix

B. Approximating the expectation of the softmax weighted sum

B.1. Motivation

We are free to choose any energy function $\tilde{E}$ in BEEBO, the only requirement being that we are able to compute an expectation of $\tilde{E}$ in order to obtain the scalar summary $E = \mathbb{E}[-\tilde{E}(\mathbf{f})]$. Of particular interest is the softmax weighted sum,

$$\tilde{E}(\mathbf{f}) = \sum_{i=1}^{Q} \text{softmax}(\beta \mathbf{f})_i f_i, \tag{8}$$

where β is the softmax inverse temperature. The softmax weight vector $\boldsymbol{\omega}$ computed as

$$\omega_i = \frac{\exp(\beta f_i)}{\sum_{j=1}^{Q} \exp(\beta f_j) + \exp(\beta_y y_{max})} \tag{9}$$

where $\exp(\beta_y y_{max})$ is an optional reference threshold value, as in expected improvement, which we set to 0 if not used (β_y is either simply β or a dynamically scaled value that ensures $\tilde{E}(\mathbf{f})$ does not become 0, see Equation 36). The parameter β allows us to interpolate between two extreme regimes,

$$\tilde{E}(\mathbf{f}) = \frac{1}{Q} \sum_{i=1}^{Q} f_i \qquad \text{for} \beta \to 0 \tag{10}$$

$$\tilde{E}(\mathbf{f}) = \max(\mathbf{f}) \qquad \text{for} \beta \to \infty \tag{11}$$

$$\tag{12}$$

In the first regime, all points of a batch equally contribute to the energy, whereas in the second regime only the single point "responsible" for the maximum is controlling the energy. Note that for numerical reasons, operating towards the $\beta \to \infty$ limit is impractical, as it will lead to zero gradients for all but one point, preventing optimization. We can set $\beta = A^{-1/2}$, with A being the prior uncertainty scale of the GP kernel. Since A represents the expected energy fluctuations for points far from data, this weighting scheme will reflect a natural compromise between Equation 11 and Equation 12.

As opposed to the mean, in the general case, the expectation of the maximum of a Q-dimensional multivariate normal is not available in closed form. To our best knowledge, this is also the case for the softmax weighted sum. In the following, we derive a closed-form approximation of the expectation of the softmax of $\beta \mathbf{f}$ that can be used for gradient-based optimization.

B.2. Derivation

Consider the softmax denominator

$$d(\mathbf{f}) = \sum_{j=1}^{Q} \exp(\beta f_j) + \exp(\beta_y y_{\text{max}}). \tag{13}$$

We will Taylor expand $\ln(d)$ to the second order, using

$$\frac{\partial \ln(d)}{\partial f_i} = \beta \frac{1}{d} \exp(\beta f_i) = \beta \omega_i \tag{14}$$

$$\frac{\partial^2 \ln(d)}{\partial f_i \partial f_j} = \beta^2 \left(-\frac{1}{d^2} \exp(\beta(f_i + f_j) + \frac{1}{d^2} \delta_{ij} \exp(\beta f_i)) \right) \tag{15}$$

$$= \beta^2 (\omega_i \delta_{ij} - \omega_i \omega_j), \tag{16}$$

where δ_{ij} is the Kronecker delta. So

$$\ln(d) \approx \ln(d_{\mathbf{a}}) + \beta \mathbf{w}^T \cdot \Delta\mathbf{f} + \frac{\beta^2}{2} \Delta\mathbf{f}^T \cdot W \cdot \Delta\mathbf{f}, \tag{17}$$

where $\mathbf{a}$ is the Taylor expansion point, $d_{\mathbf{a}}$ is d evaluated at $\mathbf{a}$, $\Delta\mathbf{f} = \mathbf{f} - \mathbf{a}$, $\mathbf{w}$ is the Q-dimensional vector $\boldsymbol{\omega}$ evaluated at $\mathbf{a}$, and W is the $Q \times Q$ matrix $W = \text{diag}(\mathbf{w}) - \mathbf{w}\mathbf{w}^T$. Inserted into Equation 9 we have

$$\omega(\mathbf{f})_i \approx w_i * \exp(\beta \Delta f_i - \beta \mathbf{w}^T \cdot \Delta\mathbf{f} - \frac{\beta^2}{2} \Delta\mathbf{f}^T \cdot W \cdot \Delta\mathbf{f}) \tag{18}$$

With this approximation we can calculate the expectation value

$$\mathbb{E}[\omega(\mathbf{f})_i * f_i] = \frac{w_i \sqrt{\det(C(\mathbf{x})^{-1})}}{(2\pi)^{Q/2}} \int \exp(\lambda_i(\mathbf{f})) * f_i df \tag{19}$$

$$\lambda_i(\mathbf{f}) = \beta \Delta f_i - \beta \mathbf{w}^T \cdot \Delta\mathbf{f}^T - \frac{\beta^2}{2} \Delta\mathbf{f}^T \cdot W \cdot \Delta\mathbf{f} - \frac{1}{2}(\mathbf{f} - \boldsymbol{\mu})^T \cdot C(\mathbf{x})^{-1} \cdot (\mathbf{f} - \boldsymbol{\mu}) \tag{20}$$

$$= c^{(i)} - \frac{1}{2}(\mathbf{f} - \boldsymbol{\nu}^{(i)})^T \cdot C(\mathbf{x})^{-1}_{\text{softmax}} \cdot (\mathbf{f} - \boldsymbol{\nu}^{(i)}), \tag{21}$$

Where $c^{(i)}$, $\boldsymbol{\nu}^{(i)}$ and $C(\mathbf{x})_{\text{softmax}}$ are defined as follows:

$$C(\mathbf{x})_{\text{softmax}} = \left(C(\mathbf{x})^{-1} + \beta^2 W\right)^{-1} \tag{22}$$

$$\mathbf{b}^{(i)} = \mathbf{e}^{(i)} - \mathbf{w} \tag{23}$$

$$\boldsymbol{\nu}^{(i)} = C(\mathbf{x})_{\text{softmax}} \cdot \left(\beta \mathbf{b}^{(i)} + C(\mathbf{x})^{-1} \cdot \boldsymbol{\mu} + \beta^2 W \cdot \mathbf{a}\right) \tag{24}$$

$$c^{(i)} = \frac{1}{2}(\boldsymbol{\nu}^{(i)})^T \cdot C(\mathbf{x})^{-1}_{\text{softmax}} \cdot \boldsymbol{\nu}^{(i)} - \beta(\mathbf{b}^{(i)})^T \cdot \mathbf{a} \tag{25}$$

$$- \frac{\beta^2}{2} \mathbf{a}^T \cdot W \cdot \mathbf{a} - \frac{1}{2} \boldsymbol{\mu}^T \cdot C(\mathbf{x})^{-1} \cdot \boldsymbol{\mu} \tag{26}$$

and $\mathbf{e}^{(i)}$ is the i'th basis vector with components $e^{(i)}_j = \delta_{ij}$. We can avoid the explict use of the precision matrix by rewriting the updated covariance matrix as

$$C(\mathbf{x})_{\text{softmax}} = \left(C(\mathbf{x})^{-1} \cdot (I + \beta^2 C(\mathbf{x}) \cdot W)\right)^{-1} = U(\mathbf{x}) \cdot C(\mathbf{x}),$$

where we have defined $U(\mathbf{x}) = \left(I + \beta^2 C(\mathbf{x}) \cdot W\right)^{-1}$. The updated mean vectors can conveniently be expressed as

$$\boldsymbol{\nu}^{(i)} = C(\mathbf{x})_{\text{softmax}} \cdot (\beta \mathbf{b}^{(i)} + C(\mathbf{x})^{-1} \cdot \boldsymbol{\mu} + \beta^2 W \cdot \mathbf{a}) \tag{27}$$

$$\boldsymbol{\nu}^{(i)} = \beta C(\mathbf{x})_{\text{softmax}} \cdot \mathbf{e}^{(i)} + C(\mathbf{x})_{\text{softmax}} \cdot \left(-\beta \mathbf{w} + C(\mathbf{x})^{-1} \cdot \boldsymbol{\mu} + \beta^2 W \cdot \mathbf{a}\right) \tag{28}$$

$$\boldsymbol{\nu}^{(i)} = \beta C(\mathbf{x})_{\text{softmax}} \cdot \mathbf{e}^{(i)} + \boldsymbol{\nu}', \tag{29}$$

where $\boldsymbol{\nu}'$ is a constant vector for all (i). Similarly

$$c^{(i)} = -\beta a_i + \frac{1}{2}\beta^2 \left(C(\mathbf{x})_{\text{softmax}}\right)_{i,i} + \beta \nu'_i + c' \tag{30}$$

$$c' = \beta \mathbf{w}^T \cdot \mathbf{a} + \frac{1}{2} \boldsymbol{\nu}'^T \cdot C(\mathbf{x})^{-1}_{\text{softmax}} \cdot \boldsymbol{\nu}' - \frac{\beta^2}{2} \mathbf{a}^T \cdot W \cdot \mathbf{a} - \frac{1}{2} \boldsymbol{\mu}^T \cdot C(\mathbf{x})^{-1} \cdot \boldsymbol{\mu} \tag{31}$$

with c' again being a constant for all (i). The expectation of the softmax weighted summary is given by

$$\mathbb{E}[\tilde{E}] = K * \sum_{i=1}^{Q} w_i * \exp(c^{(i)}) * \nu_i^{(i)} \tag{32}$$

where $K = \frac{\sqrt{\det(C(\mathbf{x})^{-1})}}{\sqrt{\det(C(\mathbf{x})^{-1}+\beta^2 W)}} = \sqrt{\det(U(\mathbf{x}))}$. The most natural choice for the expansion point is $\mathbf{a} = \boldsymbol{\mu}$ in which case $\boldsymbol{\nu}^{(i)}$ and $c^{(i)}$ reduces to

$$\boldsymbol{\nu}^{(i)} = \beta C(\mathbf{x})_{\text{softmax}} \cdot \left(\mathbf{e}^{(i)} - \mathbf{w}\right) + \boldsymbol{\mu} \tag{33}$$

$$c^{(i)} = \frac{\beta^2}{2}\left(\mathbf{e}^{(i)} - \mathbf{w}\right)^T \cdot C(\mathbf{x})_{\text{softmax}} \cdot \left(\mathbf{e}^{(i)} - \mathbf{w}\right) \tag{34}$$

B.3. Practical considerations

Linear algebra For numerical reasons, we avoid computing $C(\mathbf{x})_{\text{softmax}}$ explicitly, and instead use the $U(\mathbf{x}) \cdot C(\mathbf{x})$ factorization to compute solutions with $U(\mathbf{x})^{-1} = I + \beta^2 C(\mathbf{x}) \cdot W$.

$$C(\mathbf{x})_{\text{softmax}} \cdot \left(\mathbf{e}^{(i)} - \mathbf{w}\right) = U(\mathbf{x}) \cdot \left(C(\mathbf{x}) \cdot \mathbf{e}^{(i)} - C(\mathbf{x}) \cdot \mathbf{w}\right) \tag{35}$$

Following GPyTorch practices, we make use of the LinearOperator package to exploit the structure of $U(\mathbf{x})^{-1}$ as an `AddedDiagLinearOperator` when solving. For determinants, we find that LinearOperator's `logdet` implementation gives nondeterministic results, and we therefore perform a dense cast before computing K using default Pytorch.

While the factorization is numerically advantageous, it is still limited with regards to β. We find that at $\beta > 5$, numerical errors prevent a reliable calculation of the expectation. In practice, $A^{-1/2}$ lies in a range that allows numerically accurate solutions.

Softmax When y_{max} grows much larger than the softmax input vector $\mathbf{f}$ - a situation that can arise easily when initializing with random points for gradient-based optimization - the softmax weights $\boldsymbol{\omega}$ can become numerically zero for all "real" points, thus leading to $E(\mathbf{x}) = 0$, and vanishing gradients. As we always wish to preserve a minimal energy contribution from the real points, we parametrize the inverse temperature applied to y_{max}, β_y, using a hyperparameter α that denotes the minimal fraction of probability mass pertaining to real points.

Let N denote the softmax denominator excluding y_{max}, $N = \sum_{j=1}^{Q} \exp(\beta \Delta f_i)$. We define

$$\exp(\beta_y \Delta y_{\text{max}}) = \min\left(\frac{1-\alpha}{\alpha} N, \exp(\beta \Delta y_{\text{max}})\right) \tag{36}$$

We used $\alpha = 0.05$ as a default in all our experiments.

B.4. Relationship of BEEBO T and UCB κ hyperparameters

BEEBO bears some resemblance to the UCB acquisition function, which in the single particle mode, $Q = 1$, reads

$$a_{\text{UCB}}(x) = \mu(x) + \sqrt{\kappa}\sqrt{C(x)}, \tag{37}$$

where the parameter κ controls the balance between exploitation and exploration and $\mu(x)$ and $C(x)$ are respectively the mean and variance of the posterior distribution, $P(f \mid x, D)$, as before. We note that a_{UCB} does not account for the uncertainty of the measurement at x, and therefore remains risk-neutral under heteroskedastic noise (Makarova et al., 2021). To understand the relationship between BEEBO and UCB, we will therefore limit ourselves to the homoskedastic case and furthermore assume that measurement variances σ^2 are much smaller than the typical prior variance of the GP surrogate, A, of f, e.g. $A \simeq N^{-1}Tr(K)$, so $\sigma^2 \ll A$ and $M^{-1} = (K + \sigma^2)^{-1} \approx K^{-1}$. In this limit, the variance of $f(x)$ after measurement (indexed at $i = n$, say) reduces to σ^2:

$$C(x) = \left((K^{-1} + \sigma^{-2} I)^{-1}\right)_{nn} = \left(K(K + \sigma^2 I)^{-1}\sigma^2\right)_{nn} \approx \sigma^2 \tag{38}$$

and the information gain becomes

$$I(x) \approx \frac{1}{2}\ln(C(x)) - \log(\sigma).$$

Consequently, the gradient of the two acquisition functions reads

$$\begin{aligned}\nabla a_{\text{UCB}}(x) &= \nabla\mu(x) + \frac{\sqrt{\kappa}}{2\cdot\sqrt{C(x)}}\cdot\nabla C(x)\\ \nabla a_{\text{BEEBO}}(x) &= \nabla\mu(x) + \frac{T}{2\cdot C(x)}\cdot\nabla C(x)\end{aligned}$$

The two gradients will be identical at points x where the posterior uncertainties satisfy $\sqrt{C(x)} = \frac{T}{\sqrt{\kappa}}$. For comparison, we may desire equal gradients at iso-surfaces corresponding to a given fraction, ν, of the prior uncertainty scale $\sqrt{A}$, by setting T accordingly as $T = \nu\cdot\sqrt{A}\cdot\sqrt{\kappa}$. In our experiments, we use $\nu = \frac{1}{2}$ and configure BEEBO using a dimensionless T' explore-exploit parameter, defined as $T' = \frac{T}{\sqrt{A}}$, and set $T' = \frac{1}{2}\sqrt{\kappa}$ for a given benchmark experiment.

B.5. Relationship to GIBBON

GIBBON (Moss et al., 2021) approximates the (intractable) *General-purpose max-value Entropy Search* acquisition function, which quantifies the mutual information $MI(f^*_{\text{true}}; \mathbf{y}|D)$ of a batch of measurements $\mathbf{y}$ and the unknown optimum f^*_{true}. It does so using a lower bound on the information gain and MC estimation of the expectation over f^*_{true}. It can be written as

$$\begin{aligned}\alpha_{\text{GIBBON}}(\mathbf{x}) &= \frac{1}{2}\ln\det(R) - \frac{1}{2|\mathcal{M}|}\sum_{m\in\mathcal{M}}\sum_{i=1}^{Q}\ln\left(1-\rho_i^2\frac{\phi(\gamma_i(m))}{\boldsymbol{\phi}(\gamma_i(m))}\left[\gamma_i(m)+\frac{\phi(\gamma_i(m))}{\boldsymbol{\phi}(\gamma_i(m))}\right]\right)\\ \alpha_{\text{GIBBON}}(\mathbf{x}) &= \frac{1}{2}\ln\det(R) + \sum_{i=1}^{Q}\hat{\alpha}_{\text{GIBBON}}(x_i),\end{aligned} \tag{39}$$

where R is the correlation matrix with entries $R_{ij} = \frac{C(\mathbf{x})_{ij}}{\sqrt{C(\mathbf{x})_{ii}C(\mathbf{x})_{jj}}}$, $\mathcal{M}$ is a set of samples for the max-value f^*_{true}, and ρ_i is the correlation of y_i and $f_{\text{true}}(x_i)$. $\boldsymbol{\phi}$ and ϕ are the standard normal cumulative distribution and probability density functions, and $\gamma_i(m) = \frac{m-\mu}{\sigma}$.

The definition of BEEBO introduced in Equation 7, with the scalar summarization function set to the expected mean, $E(\mathbf{x}) = -\frac{1}{Q}\sum_{i=1}^{Q}\mu(x_i)$, gives

$$\alpha_{\text{BEEBO}}(\mathbf{x}) = T * \frac{1}{2}\left(\ln\det C(\mathbf{x}) - \ln\det C_{\text{aug}}(\mathbf{x})\right) + \sum_{i=1}^{Q}\mu(x_i). \tag{40}$$

From the second formulation of GIBBON, it becomes obvious that although being distinct in their motivation and derivation, BEEBO and GIBBON implement acquisition functions with a similar structure. Taking an information theoretic and multi-fidelity BO standpoint, GIBBON refers to this trade-off as *diversity* against *quality*, whereas in BEEBO we follow the intuitions of UCB, and use *exploration* and *exploitation*.

- *Quality - Exploitation*: GIBBON employs an MC estimate of the lower bound approximation of the information gain provided by each point, whereas BEEBO directly summarizes the optimality of all points in closed form, either as their mean or an approximated softmax weighted sum.
- *Diversity - Exploration*: In GIBBON, the diversity derived from the differential entropy $H(\mathbf{f}|D,\mathbf{x})$ is the entropy of the posterior correlation $\frac{1}{2}\ln\det(R)$. In BEEBO, we employ the *reduction* of entropy, the information gain $I(\mathbf{x})$. Under homoskedastic noise, $I(\mathbf{x}) \propto \ln\det(C(\mathbf{x}))$. Since $R(\mathbf{x}) = \text{diag}(C(\mathbf{x}))^{-1/2}\cdot C(\mathbf{x})\cdot\text{diag}(C(\mathbf{x}))^{-1/2}$, we have that $\ln\det(R) = \ln\det(C(\mathbf{x})) - \sum_i^Q\ln(C(\mathbf{x})_{ii})$. Therefore, maximizing the log determinant of R penalizes points that have high variance.

Therefore, while GIBBON presents an attractive approximation of max-value Entropy Search for batched acquisition, BEEBO is an alternative that avoids approximating a quality criterion using MC. Moreover, GIBBON's diversity criterion implicitly penalizes points that have high variance, whereas BEEBO's criterion maximizes the reduction of variance.

In the context of large batches ($Q >> 10$), a modification of GIBBON exists that is further similar to BEEBO. Departing from the strict max-value entropy search derivation, a scaling factor Q^{-2} is introduced to counteract a growing dominance of the diversity term:

$$\alpha_{\text{scaledGIBBON}}(\mathbf{x}) = \frac{1}{2Q^2} * \ln\det(R) + \sum_{i=1}^{Q} \hat{\alpha}_{\text{MES}}(x_i) \tag{41}$$

This scaling is motivated by the fact that R contains Q^2 elements. However, we note that R is summarized by its log determinant, which scales linearly in Q: As the determinant is the product of the eigenvalues, the log determinant is the sum of the log-eigenvalues. The number of eigenvalues scales linearly with matrix size Q, and so does the log determinant.

C. Implementation details

C.1. Acquisition function optimization

BEEBO was implemented for full compatibility with the BoTorch framework (version 0.9.4) (Balandat et al., 2020) as an `AnalyticAcquistionFunction`. Standard BoTorch utilities for initializing and training GPs, initializing q-batches and performing gradient descent optimization of the acquisition function are used. We trained GPyTorch (version 1.11) (Gardner et al., 2018) GP models with KeOps (Charlier et al., 2021) Matérn 5/2 kernels (following BoTorch defaults with a separate length scale for each input dimension, and Gamma priors on the length and output scales). Log determinants for the information gain were computed using singular value decomposition for numerical stability.

GPyTorch provides a `get_fantasy_model` method that allows for the efficient augmentation of the training data of a GP with a set of points, as done in BEEBO. However, we observed that GPyTorch's implementation suffers from GPU memory leaks when used with automatic differentiation enabled. We therefore instantiate augmented models explicitly, not making use of the (more efficient) augmentation strategy.

All experiments were performed with double precision. `SobolQMCNormalSampler` was used for acquisition functions making use of the reparametrization trick.

C.2. Benchmark BO methods

All methods were benchmarked in BoTorch. For q-EI, we used LogEI (Ament et al., 2023). For TS, 10.000 base Sobol samples were drawn and sampled with `MaxPosteriorSampling` using the Cholesky decomposition of the covariance matrix. GIBBON was optimized using sequential optimization following the BoTorch tutorial. We additionally implemented a custom version of GIBBON that applies the Q^{-2} scaling factor to the diversity term, as proposed in GIBBON's supplementary material. We used 100,000 random discretized candidates for max-value sampling. In a few iterations, optimizing GIBBON seemed challenging, with BoTorch reporting that no nonzero initialization candidate could be identified. KB was optimized using a custom greedy optimization loop with fantasized observations, using (single-sample) LogEI as the underlying acquisition function. None of the methods use a hyperparameter for controlling their explore-exploit trade-off. The results are therefore based on 10 iterations at defaults.

C.3. Test problems

Test functions All test functions were used in their BoTorch implementations. As done in previous work, the embedded Hartmann function was created by appending all-0 dummy dimensions to the original six dimensions (Wang et al., 2016; Papenmeier et al., 2022; Eriksson & Jankowiak, 2021). In round 0, we started optimization with 100 random points sampled at 0.5 minimum distance to the problem's true optimum.

D. Extended results

D.1. Results including additional baselines

Table A1. BO on noise-free synthetic test problems. The normalized highest observed value after 10 rounds of BO with q=100 is shown. Colors are normalized row-wise. The BEEBO and q-UCB columns are equivalent to Table 1. Higher means better. Results are means over five replicate runs.

Problem	d	meanBEEBO			maxBEEBO			q-UCB			q-EI	TS	KB	GIBBON	
		T'=0.05	T'=0.5	T'=5.0	T'=0.05	T'=0.5	T'=5.0	$\sqrt{\kappa}$=0.1	$\sqrt{\kappa}$=1.0	$\sqrt{\kappa}$=10.0	-	-	-	default	scaled
Ackley	2	0.993	0.975	0.972	0.976	0.965	0.982	0.961	0.973	0.996	0.991	1	0.969	0.889	0.965
Levy	2	1	1	1	1	1	0.998	1	0.999	0.997	1	0.999	1	0.991	0.988
Rastrigin	2	0.977	0.988	0.974	0.996	0.983	0.991	0.938	0.982	0.935	0.995	1	0.987	0.878	0.937
Rosenbrock	2	0.992	0.962	0.98	0.998	0.975	0.975	0.949	0.905	0.984	0.985	0.987	0.997	0.699	0.905
Styblinski-Tang	2	0.962	1	1	1	1	1	1	1	0.999	1	1	1	0.995	0.999
Shekel	4	0.509	0.902	0.755	0.327	0.336	0.418	0.284	0.391	0.266	0.377	0.159	0.353	0.221	0.302
Hartmann	6	1	1	0.978	0.896	0.974	0.967	0.91	0.957	0.895	0.992	0.807	0.988	0.87	0.906
Cosine	8	1	0.999	0.644	0.999	0.975	0.929	0.94	0.926	0.716	0.801	1	0.984	0.88	0.918
Ackley	10	0.913	0.888	0.828	0.829	0.726	0.543	0.807	0.775	0.52	0.802	1	0.766	0.402	0.563
Levy	10	0.989	0.964	0.953	0.964	0.945	0.911	0.885	0.898	0.606	0.915	0.955	0.975	0.882	0.862
Powell	10	0.987	0.966	0.899	0.952	0.954	0.931	0.908	0.909	0.256	0.924	0.91	0.972	0.746	0.817
Rastrigin	10	0.483	0.503	0.598	0.532	0.512	0.606	0.413	0.537	0.299	0.441	1	0.411	0.367	0.295
Rosenbrock	10	0.993	0.993	0.929	0.992	0.987	0.971	0.965	0.966	0.71	0.987	0.866	0.994	0.976	0.982
Styblinski-Tang	10	0.797	0.812	0.242	0.784	0.643	0.217	0.233	0.368	0.072	0.689	0.442	0.808	0.027	0.266
Ackley	20	0.843	0.857	0.789	0.819	0.788	0.39	0.744	0.753	0.566	0.739	1	0.759	0.25	0.344
Levy	20	0.936	0.939	0.896	0.953	0.901	0.911	0.92	0.899	0.786	0.937	0.979	0.949	0.903	0.693
Powell	20	0.947	0.966	0.84	0.936	0.88	0.908	0.951	0.905	0.819	0.913	0.964	0.964	0.893	0.867
Rastrigin	20	0.373	0.462	0.518	0.514	0.48	0.491	0.431	0.503	0.401	0.467	1	0.495	0.405	0.459
Rosenbrock	20	0.993	0.993	0.906	0.991	0.982	0.923	0.973	0.979	0.911	0.983	0.939	0.995	0.966	0.977
Styblinski-Tang	20	0.706	0.669	0.305	0.607	0.417	0.279	0.187	0.542	0.029	0.607	0.266	0.621	0.094	0.315
Ackley	50	0.221	0.146	0.842	0.622	0.705	0.457	0.622	0.758	0.732	0.722	1	0.565	0.291	0.119
Levy	50	0.976	0.978	0.943	0.977	0.955	0.867	0.951	0.955	0.933	0.952	0.987	0.956	0.867	0.905
Powell	50	0.94	0.978	0.959	0.976	0.97	0.929	0.97	0.953	0.978	0.962	0.985	0.941	0.953	0.931
Rastrigin	50	0.273	0.505	0.453	0.473	0.466	0.445	0.446	0.405	0.501	0.47	1	0.376	0.442	0.331
Rosenbrock	50	0.979	0.984	0.988	0.978	0.981	0.968	0.972	0.987	0.987	0.978	0.979	0.988	0.982	0.981
Styblinski-Tang	50	0.605	0.693	0.417	0.536	0.415	0.332	0.341	0.713	0.416	0.702	0.254	0.713	0.224	0.529
Ackley	100	0.31	0.347	0.864	0.526	0.536	0.707	0.711	0.634	0.848	0.73	0.007	0.284	0.399	0.341
Emb. Hartmann 6	100	0.98	0.988	0.916	0.982	0.933	0.914	0.894	0.923	0.921	0.902	0.554	0.947	0.708	0.715
Levy	100	0.89	0.966	0.943	0.962	0.942	0.946	0.954	0.938	0.965	0.954	0.31	0.916	0.868	0.865
Powell	100	0.795	0.946	0.987	0.985	0.981	0.981	0.982	0.978	0.984	0.965	0.406	0.964	0.695	0.6
Rastrigin	100	0.522	0.367	0.467	0.481	0.479	0.469	0.437	0.453	0.443	0.481	0.238	0.626	0.232	0.297
Rosenbrock	100	0.812	0.975	0.975	0.928	0.978	0.975	0.97	0.966	0.983	0.975	0.27	0.941	0.989	0.933
Styblinski-Tang	100	0.564	0.47	0.396	0.432	0.331	0.309	0.307	0.574	0.289	0.559	0.198	0.619	0.223	0.221

Table A2. BO on noise-free synthetic test problems. The relative batch instantaneous regret of the last, exploitative batch is shown. Colors are normalized row-wise. The BEEBO and q-UCB columns are equivalent to Table 2. Lower means better. Results are means over five replicate runs.

Problem	d	meanBEEBO			maxBEEBO			q-UCB			q-EI	TS	KB	GIBBON	
		T'=0.05	T'=0.5	T'=5.0	T'=0.05	T'=0.5	T'=5.0	$\sqrt{\kappa}$=0.1	$\sqrt{\kappa}$=1.0	$\sqrt{\kappa}$=10.0	-	-	-	default	scaled
Ackley	2	0.28	0.23	0.25	0.25	0.24	0.14	1.01	0.99	1.01	0.93	0.56	0.79	1.04	0.82
Levy	2	0.13	0.09	0.09	0.12	0.09	0.11	1.38	0.99	1.16	1.17	0.23	0.16	1.58	0.56
Rastrigin	2	0.43	0.44	0.51	0.66	0.48	0.53	1.03	1.00	1.04	0.80	0.68	0.72	1.21	0.67
Rosenbrock	2	0.00	0.00	0.00	0.00	0.00	0.00	0.86	0.92	0.87	1.20	0.00	0.00	0.73	0.07
Styblinski-Tang	2	0.17	0.17	0.17	0.17	0.17	0.17	1.01	1.02	1.06	0.89	0.04	0.46	2.04	1.00
Shekel	4	0.84	0.66	0.68	0.75	0.72	0.67	0.99	0.99	0.99	0.97	0.93	1.00	1.00	0.98
Hartmann	6	0.05	0.08	0.10	0.22	0.10	0.11	0.93	0.97	0.86	0.88	0.36	0.44	1.03	1.02
Cosine	8	0.00	0.00	0.23	0.00	0.02	0.05	0.95	0.96	0.90	1.16	0.45	0.98	1.64	0.55
Ackley	10	0.42	0.32	0.24	0.34	0.32	0.46	0.93	0.94	0.94	0.94	0.98	0.88	1.02	1.01
Levy	10	0.04	0.02	0.23	0.03	0.04	0.10	1.26	0.97	1.14	0.74	0.51	0.43	2.78	1.46
Powell	10	0.01	0.01	0.06	0.03	0.07	0.14	1.01	1.06	1.23	0.24	0.16	0.04	2.90	0.63
Rastrigin	10	0.64	0.48	0.59	0.46	0.51	0.45	0.91	0.89	0.90	0.94	0.84	0.97	1.19	0.76
Rosenbrock	10	0.00	0.00	0.07	0.01	0.01	0.05	0.91	0.77	0.93	0.09	0.10	0.00	2.49	0.19
Styblinski-Tang	10	0.21	0.23	0.58	0.23	0.33	0.49	1.26	1.12	1.23	1.32	0.82	0.69	2.37	0.98
Ackley	20	0.67	0.26	0.21	0.22	0.31	0.61	0.95	0.95	0.91	0.92	0.98	1.02	1.02	0.85
Levy	20	0.08	0.08	0.12	0.18	0.11	0.21	0.86	0.90	1.04	1.40	0.74	0.21	3.32	1.13
Powell	20	0.10	0.01	0.03	0.08	0.08	0.12	0.87	0.70	0.92	0.09	0.47	0.02	3.39	0.25
Rastrigin	20	0.71	0.61	0.51	0.62	0.64	0.56	0.86	0.85	0.85	0.81	0.86	0.87	1.28	0.65
Rosenbrock	20	0.05	0.00	0.04	0.12	0.05	0.05	0.59	0.61	0.94	0.05	0.40	0.01	2.66	0.24
Styblinski-Tang	20	0.40	0.36	0.73	0.40	0.53	0.56	1.10	1.14	1.16	0.99	0.89	0.83	2.82	1.11
Ackley	50	0.90	0.86	0.16	0.40	0.47	0.54	0.94	0.94	0.87	0.83	0.99	0.65	1.01	0.91
Levy	50	0.04	0.03	0.04	0.02	0.05	0.24	0.61	0.69	0.95	0.14	0.87	0.08	1.89	0.17
Powell	50	0.02	0.02	0.01	0.02	0.04	0.08	0.42	0.54	0.81	0.06	0.88	0.03	0.25	0.06
Rastrigin	50	0.75	0.60	0.89	0.59	0.59	0.58	0.80	0.79	0.76	0.68	0.93	0.94	1.29	0.73
Rosenbrock	50	0.02	0.01	0.01	0.01	0.04	0.05	0.46	0.47	0.61	0.04	0.81	0.01	1.17	0.03
Styblinski-Tang	50	0.46	1.06	0.71	0.45	0.57	0.72	0.97	1.18	0.99	0.65	0.96	0.76	1.46	0.61
Ackley	100	0.68	0.82	0.14	0.55	0.48	0.29	0.95	0.94	0.87	0.80	1.00	0.77	1.01	0.93
Emb. Hartmann 6	100	0.09	0.04	0.18	0.14	0.10	0.19	0.57	0.88	0.72	0.51	0.88	0.14	0.87	0.88
Levy	100	0.09	0.04	0.04	0.04	0.17	0.06	0.63	0.72	0.60	0.14	0.98	0.09	0.88	0.16
Powell	100	0.11	0.03	0.01	0.01	0.03	0.01	0.43	0.54	0.58	0.12	1.02	0.03	0.42	0.48
Rastrigin	100	0.50	0.58	0.55	0.47	0.55	0.55	0.76	0.83	0.77	0.66	0.99	0.63	0.83	0.69
Rosenbrock	100	0.12	0.02	0.02	0.06	0.06	0.04	0.53	0.62	0.54	0.11	0.98	0.04	0.86	0.12
Styblinski-Tang	100	0.37	0.45	0.51	0.55	0.63	0.77	0.91	1.20	0.95	0.48	0.99	0.34	0.80	0.87