Parallel cost-aware optimization of multidimensional black-box functions

Oliver Sihlovec

Supervisor(s): Matthijs Spaan[1], Joery de Vries[1]

[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 25, 2023

Name of the student: Oliver Sihlovec
Final project course: CSE3000 Research Project
Thesis committee: Matthijs Spaan, Joery de Vries, Christoph Lofi

An electronic version of this thesis is available at http://repository.tudelft.nl/.

Abstract

Scientific problems are often concerned with optimization of control variables of complex systems, for instance hyperparameters of machine learning models. A popular solution for such intractable environments is Bayesian optimization. However, many implementations disregard dynamic evaluation costs associated with the optimization procedure. Furthermore, another common trope among Bayesian algorithms is that they are short-sighted and do not consider long-term effects of their actions. This paper investigates the viability of multi-timestep cost-aware Bayesian optimizers and evaluates their performance in environments with delayed rewards. To this end, we combine existing works on parallel Bayesian optimizers and cost-aware heuristics. Our findings reveal that although such parallel optimizers yield more optimal results and are more resistant to delayed feedback compared to their myopic counterparts, they are unable to achieve cost-awareness.

1 Introduction

Bayesian Optimization [1] is an algorithmic technique designed for optimization of expensive to evaluate, black-box functions. Such functions commonly occur in domains such as Hyperparameter Optimization [2] or Algorithm Selection [3], wherein evaluating the cost function is expensive in terms of time and or resources. Bayesian optimizers find near global optimums by iteratively estimating the posterior distribution of unknown function values for desired set of points given the priors, as parameterized by previously evaluated data. These proxies are then used to optimize for a utility metric correlated to the actual fitness of the objective function. As a result, the optimization is shifted from the real world to a surrogate space that is significantly cheaper to evaluate.

An extension of the aforementioned problem is multi-objective Bayesian optimization, where the goal is to determine the Pareto set of the objective functions. This set is composed of all parameter combinations that are not a strict improvement over any other combination with respect to the objectives.

For both single and multi objective variants, Bayesian optimizers typically implement an acquisition function, which outputs promising data points that are likely to yield near optimal values, or exploratory points that reveal information about the functional landscape. This reduces the number of function invocations and avoids searching the objective space using brute-force.

However, current approaches assume that the cost of function evaluation is uniform in terms of the input, which is not necessarily true in practice. As an example, consider the case of optimizing the architecture of neural networks, where increasing the number of neurons per layer leads to longer training times, as the machine learning algorithm needs to fine-tune more weights.

Abdolshah et al. [4] propose a multi-objective cost-aware acquisition function that supplements the scalarized Upper Confidence Bound (UCB) acquisition function with cost-aware heuristic that encourages evaluation of inexpensive inputs. The underlying acquisition function, however, suffers from two main drawbacks. Firstly, it is one of the simplest acquisition functions and is thus sensitive to the hyperparameter pertaining to exploration versus exploitation trade-off [5]. Secondly, UCB is short-sighted when subjected to delayed feedback, therefore only performing single-step optimization and as a result does not plan out future actions. This particular drawback is especially significant, as it can have a considerable impact on the results of optimization for such ill-defined environments[6].

An alternative to UCB is the Expected Improvement (EI) that leverages the likelihood of improvement and the associated potential gain, which is more robust to hyperparameter selection. Unfortunately, it is greedy by design and is thus just as myopic as UCB. To this end, Wang et al. [7] introduce an extension (qEI), which maximizes the expected improvement across a set of multiple points, hence taking future evaluations into account. Moreover, since the evaluations are performed in parallel and the algorithm is compatible with both synchronous and asynchronous environments, it is no slower than the standard EI with respect to wall-clock speed.

Thus, the goal of this paper is to answer the following research questions:

1. Does parallelization of the acquisition function enhance the optimality of multi-objective Bayesian optimizers?
2. Can parallel expected improvement (qEI) be adapted in order to achieve cost-awareness for multi-objective Bayesian optimization?
3. Is the multi-timestep variant of cost-aware Bayesian optimization of multiple objectives resilient to environments with delayed feedback?

2 Related Work

To the best of our knowledge, no existing research investigates multi-objective cost-aware Bayesian optimization from the perspective of parallel acquisition functions. However, there do exist various works that explore the building blocks underpinning this paper.

Lee et al. [8] propose a method for parallel cost-aware Bayesian optimization of a single objective. Their algorithm also penalizes expensive inputs during the early stages and gradually relaxes this constraint. Unlike [4], their approach does not assume an a priori set of cost preferences and is specifically tailored for problems concerning hyperparameter optimization. Instead, they utilize specialized algorithms that infer the cost by counting the flops of the model's subroutines. This approach is thus more suitable for black-box settings, wherein the user does not possess any prior knowledge about the system.

Guinet et al. [9] also introduce an algorithm for cost-aware hyperparameter optimization using Bayesian optimizers. Re-

markably, they treat the cost as a separate objective and attempt to find the Pareto frontier thereof and the original objective adjusted by a scaling parameter for each iteration of the algorithm. As a result, varying the scaling parameter controls the cost-awareness of the optimizer. Similarly to the work of Lee et al. [8], the authors also adopt the same models for cost inference.

Finally, Daulton et al. [10] present a method for parallel Bayesian optimization of multiple objective functions. More specifically, their acquisition function directly optimizes for the improvement of dominated hypervolume [11]. This contrasts scalarizitation of the individual objectives that transforms multi-objective optimization to a single-objective setting. However, this approach also condenses multiple objectives into a scalar value representing the fitness of a sample, but does so at a different stage. Last but not least, the authors design this acquisition function particularly for noisy environments that considerably distort the output values of the objectives.

3 Background

This section outlines the theoretical framework necessary for defining the problem and methodology in the subsequent sections.

Bayesian Optimization

Bayesian optimization aims to determine the global optimum of a black-box objective function. In this paper, the global maximum is henceforth considered in place of global optimum. Thus, Bayesian optimizers seek to find:

$$\max_{x \in X} f(x) \tag{1}$$

where f denotes the objective function and $X \subseteq R^d$ represents the input space of dimensionality d. Function f is further assumed to be noisy, expensive to evaluate, non-differentiable and with unknown analytical expression. As a result, Bayesian optimizers approximate the posterior distribution of unknown datapoints given a set of previously evaluated inputs and attempt to find the maximum thereof.

Multi-Objective Optimization

The aforementioned problem can be extended to multidimensional objective functions. Supposing an m-dimensional objective function $f : R^d \rightarrow R^m$, a point $y = \{y_1, .., y_m\}$, is said to be Pareto optimal if and only if there exists no other point $y^* = \{y_1^*, .., y_m^*\}$ for which $y_1^* \geq y_1, .., y_m^* \geq y_m$, with at least one inequality being strict. Otherwise, y^* is said to dominate y, which is denoted by $y^* \succ y$. The goal in this setting is to determine the set of inputs that map to the Pareto optimal front over the objective landscape. This can be achieved using Chebyshev's scalarization, that is, finding the minimum of the scalarized multi-objective function:

$$\operatorname*{argmax}_{x \in R^d} \min(w_1 f_1(x), .., w_m f_m(x)) \tag{2}$$

where w_i are the weights of the respective objectives, whose variations facilitate modelling of the Pareto front.

Gaussian Process

A collection of random variables is said to follow Gaussian Process if and only if each of its finite subsets is normally distributed. Assuming that the black-box function follows the Gaussian Process, denoted as

$$f \sim GP(\mu, k) \tag{3}$$

then the distribution of known function outputs for a collection of inputs X is given by

$$f(X) \sim N(\mu(X), \Sigma(X)) \tag{4}$$

where $\mu_i = \mu(X_i)$ represents the mean of point i and $\Sigma_{ij} = k(X_i, X_j)$ represents the noise given by the covariance between points i and j. It can as a result be shown [12, pp. 13-16] that the posterior distribution of an unknown set of datapoints X^* is estimated according to:

$$f(X^*)|X^*, f(X), X \sim N(\mu(X^*), \Sigma(X^*)) \tag{5}$$

where

$$\mu(X^*) = \mu(X) + k(X^*, X)k(X, X)^{-1}(f(X) - \mu(X))$$

$$\Sigma(X^*) = k(X^*, X^*) - k(X^*, X)k(X, X)^{-1}k(X, X^*)$$

Acquisition Function

In order to avoid querying random inputs, Bayesian optimizers utilize acquisition functions that strategically select the next datapoint which is likely to yield an improvement. Furthermore, acquisition functions are commonly equipped with parameters controlling the exploration versus exploitation trade-off. That is, rather than selecting a supposedly slightly more optimal input, an acquisition function may instead decide to explore regions with high uncertainty. An example of an acquisition function is the Upper Confidence Bound, defined as follows:

$$x_{t+1}^* = \operatorname*{argmax}_{x \in R^D}(\mu_t(x) + \sqrt{\beta}\sigma_t(x)) \tag{6}$$

where x_{t+1}^* denotes the next promising input, $\mu_t(x)$ denotes the expected point x's output value, as predicted by the current fit and $\sigma_t(x)$ represents its standard deviation. Parameter $\sqrt{\beta}$ controls the exploitative or exploratory behaviour of the optimizer.

An alternative to UCB is Multi-points Expected Improvement (qEI) that maximizes the improvement across a batch of q points:

$$X_{t+1}^* = \operatorname*{argmax}_{X \in R^{d \times q}} E[(\max_{i=1,..,q} f(X_i) - f(x_{max}))^+] \tag{7}$$

where X_{t+1}^* denotes the next batch of promising inputs, $f(x_{max})$ denotes the current global maximum and X_i represents the i-th point of batch X, with $(x)^+ = \max(x, 0)$. The expectation is assumed to be non-negative, as querying a sub-optimal batch of points would yield no improvement.

Cost-Awareness

In the context of this research, we define the cost as a secondary objective separate to those encapsulated in multi-objective optimization. However, taking the cost into account can indirectly optimize for the primary objectives. For instance, it can facilitate more evaluations of the objective functions under the assumption that the cost corresponds to the time of execution. Towards this end, we follow the verbatim definition of cost-aware constraints by Abdolshah et al. [4]:

Definition 1 (Cost-Aware Constraints)**.** Let $I = (i_1, i_2, ..., i_k) | \{i_1, i_2, ..., i_k\} \subset \mathbb{Z}_d^+, i_j \neq i_{j'} \forall j \neq j'$ be a cost-aware constraint over k dimensions of the search space ($1 \leq k \leq d$). Then selecting $x_{(i_j)}$ value as a input from dimension i_j of the search space is more expensive than selecting the same value of x from dimension i_{j+1} of the search space given that $x_{1,...,k}$ are in the same normalized range and $j \in \mathbb{Z}_{k-1}^+$.

As a result, this definition can be understood as specifying the relative cost of a subset of features based on a sorted tuple of indices [4]. Based on this definition, the authors propose the following heuristic that assigns the cost to a given input vector at timestep t:

$$C(x, t) = \prod_{j=1}^{k} (1 - \pi(x_{I_{(j)}}, t)) \tag{8}$$

$$\pi(x_{I_{(j)}}, t) = Exp(x_{I_{(j)}}, \lambda) = \lambda e^{-\lambda x}, \lambda = \frac{1}{\theta_{I_{(j)}} t + 1} \tag{9}$$

where $\pi(x_{I_{(j)}}, t)$ follows the exponential distribution, with respective weights $\theta_{I_{(j)}}$ being sampled from Dirichlet's distribution $\theta_{I_{(1)}}, ..., \theta_{I_{(k)}} \sim Dir(1, ..., 1)$ and sorted descendingly as given by I.

4 Methodology

This section is divided into three subsections, first of which is dedicated to the explanation of our algorithm. The second subsection describes the simulation of an environment with delayed rewards. The final subsection explains the experimental setup, whose results will be reviewed in the following sections.

Algorithm

Abdolshah et al. [4] combine the concepts of multi-objective optimization, upper-confidence bound (UCB) and cost-awareness into the following acquisition function:

$$\alpha(x, W_t, t) = Q(x, W_t) \times (1 - C(x, t)) \tag{10}$$

where $Q(x, W_t)$ refers to the UCB acquistion function (6), whose Gaussian process estimates the scalarized utility given by (2). The Chebyshev scalarization is weighted by normalized weights W_t drawn from a uniform distribution. The second term $C(x, t)$ refers to the cost-aware constraints (8).

We propose an adjustment to the aforementioned acquisition function, such that we substitute parallel expected improvement (7) for the UCB (6). Furthermore, since the cost heuristic is not defined for a batch of points, we aggregate the cost by taking the average of the individual costs per point in the batch:

$$C(X, t) = \frac{1}{q} \times \sum_{i=1}^{q} C(X_i, t) \tag{11}$$

As a result, we implement Bayesian Optimization for modelling of the Pareto front by varying the weights W_t at every timestep t and utilizing the Gaussian Process regressor for estimating the output space of each objective. Our adjusted acquisition function then maximizes the expected improvement of the scalarization of the primary objectives, for a batch of points at timestep $t + 1$, whilst penalizing expensive regions of the input space. For more details, the implementation is outlined in the pseudocode that can be found in the appendix.

Simulation of Delayed Environment

For the purpose of this paper, we simulate the delay by halting the objective outputs, storing them in a temporary buffer and forwarding a dummy output to the optimizer. The delay is controlled by parameter d, which corresponds to the response period. That is, the optimizer receives a response every d timesteps, which contains the true objective outputs of d previous queries.

In practice, there are many strategies that the optimizers can adopt to combat the delay. Based on the previous definition of delay, the optimizers studied in this paper impute the suppressed output values as predicted by their internal Gaussian process regressors. While the actual evaluations are unavailable, the queried datapoints are mapped to these predictions. The optimizers then update their observations accordingly as soon as the objective outputs are released. Formally, we can represent this as:

$$Y_{ij}^t = f_i(x_j) \text{ if } t - j \geq d \text{ or } t \equiv 0 \pmod{d}$$

$$Y_{ij}^t \sim N(\mu_i(x_j), \Sigma_i(x_j)) \text{ otherwise}$$

where Y^t is an $m \times t$ matrix containing the observed output of objective i for query point x_j at timestep j, t denotes the current timestep, f_i is the i-th objective function and $N(\mu_i(x_j), \Sigma_i(x_j))$ follows the definition of Gaussian Process from equation (5).

Lastly, we adapt our implementation of qEI such that the batch of points returned by the acquisition function at each timestep is queried sequentially. As a result, the optimizer does not need to impute the missing data as long as the entirety of the batch has not been processed. One consequence of this method is that depending on the choice of d and q, the degree of parallelism, the optimizer might have access to the objective values for some inputs of the previous batch, while having to impute the outputs for the rest.

Experimental Setup

The evaluation consists of two experiments. The first experiment serves as a proof of concept, exploring performance of cost-aware qEI (CA-qEI) in a correctly specified environment. The second experiment evaluates the impact of an

environment with delayed feedback on optimality and cost-awareness of CA-qEI. In both settings, our method is benchmarked against the baselines, the default cost-unaware qEI (simply qEI) and cost-aware UCB (CA-UCB) [4], respectively. For completeness, an overview of the performance of a random strategy in both experiments can be found in the appendix.

In **both** settings, the algorithm is going to be evaluated both in terms of optimization performance as well as cost-awareness. The former will be measured using the dominated hypervolume [11]. This metric is useful for quantifying the fitness of a Pareto front with respect to a given reference point, being proportional to the optimality of the given solution. On the other hand, cost-awareness is going to be measured by assessing the difference of cumulative sums of L1 norms per each dimension of the input space. Aboldshah et al. [4] used this metric, as cost-unaware implementations are positively correlated with a more uniform distribution of the expenses per dimension, hence the difference being negligible or possibly negative and vice-versa for their counterparts.

For the **first** experiment, we assume a budget of fifty timesteps and eight-fold parallelization. Cross-in-tray [13] and Hölder table functions [14] constitute the two objectives, which we seek to optimize on interval $[-10, 10]$ for both dimensions of the input space. With x representing the 2-dimensional input vector, the two functions are respectively defined as:

$$f_1(x) = 0.0001 \times \left| \sin(x_1)\sin(x_2)\exp\left|100 - \frac{\sqrt{x_1^2 + x_2^2}}{\pi}\right| + 1\right|^{0.1}$$

$$f_2(x) = \left| \sin(x_1)\cos(x_2)\exp\left|1 - \frac{\sqrt{x_1^2 + x_2^2}}{\pi}\right|\right| \tag{13}$$

In the **second** experiment, we also assume a budget of 50 iterations. Furthermore, we reinforce robustness by varying the delay parameter, which takes values of 5, 10 and 25. We also vary the degree of parallelization, which corresponds to 8 in the first setting and is then increased to 16 for the latter two. Finally, we utilize the shifted Drop-wave function [13] and Rastrigin function [15] on interval $[-5, 5]$ for both dimensions, as the two objectives. With x once again representing the 2-dimensional input vector, these functions are respectively defined as follows:

$$f_1(x) = 2\frac{1 + \cos\left(12\sqrt{(x_1 + \pi)^2 + (x_2 - \pi)^2}\right)}{(x_1 + \pi)^2 + (x_2 - \pi)^2 + 2}$$

$$f_2(x) = 20 + \sum_{i=1}^{2} x_i^2 - 10\cos(2\pi x_i) \tag{14}$$

5 Results

Figures 1a through 1f showcase the results pertaining to the **first** experiment. We can firstly observe that the dominated hypervolume has converged by the end of the experiment, aside for a small deviation in case of cost-unaware qEI. Although CA-qEI was constrained by the cost of inputs, it has achieved dominated hypervolume of 0.7202, which is slightly more optimal than the score of the default qEI of 0.6427. Meanwhile, cost-aware UCB only attained 0.5630 for the same metric, hence being inferior to both variants of qEI and only barely outperforming random strategy with score of 0.4769, whose plot can be found in the appendix.

However, only UCB has managed to accomplish cost-awareness, where the disparity between sums of selected inputs became increasingly more apparent with each timestep. Both implementations of qEI on the other hand have in fact prioritized the more expensive dimension, although the difference is insignificant compared to CA-UCB. Interestingly, CA-qEI appears to have actually been more agnostic to the relative costs than standard qEI, albeit by a tiny margin.

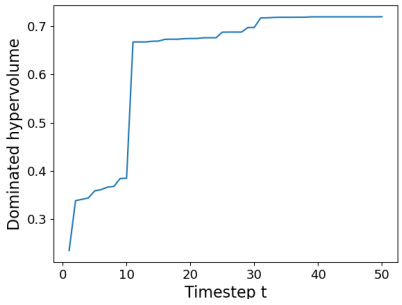
Figures 2a through 2f depict the results observed in the **second** experiment[1]. Once again, the dominated hypervolume has stabilized near the final timesteps for all settings. Additionally, we can see that CA-qEI and regular qEI outperform CA-UCB when operating with higher batch sizes, peaking at 0.4986 and 0.5081 respectively in dominated hypervolume for delay of 10 timesteps and batch size of 16. On the other hand, both variants of qEI were unable to outperform CA-UCB when having prepared 8 inputs in advance. Meanwhile, CA-UCB has converged to a dominated hypervolume of approximately 0.3 in all three settings. However, it is important to remark that the random strategy. detailed in the appendix, resulted in a dominated hypervolume of 0.2372, which is very competitive with all strategies except for qEI and CA-qEI for one of the configurations.

Similarly to the first experiment, CA-qEI and baseline qEI struggle to maintain cost-awareness, regardless of the setting, which can be observed from the rapid oscillations of cost differences of selected outputs. However, CA-UCB was able to swiftly establish a foothold and continued to widen the gap with each timestep. Its cost-awareness was most prevalent when subjected to a delay of 5 timesteps, amounting to a cost difference of 56.0938. The performance in the remaining settings was very comparable, with a delay of 10 timesteps resulting in a final cost difference of 27.8347, while a delay of 25 timesteps resulted in a final cost difference of 28.7093.

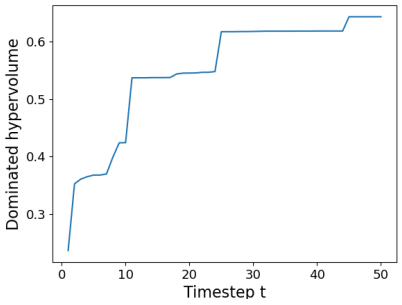
6 Discussion

Firstly, it is important to emphasize that qEI is more successful than UCB in terms of multi-objective optimization, as evidenced by all experiments, which we believe can be attributed to the ability of performing significantly more queries in the commensurate number of timesteps. This supports the hypothesis that parallelization can refine serial implementations

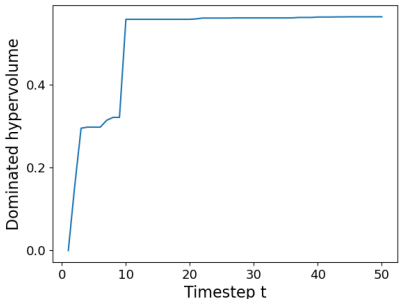
[1]Since CA-UCB is a non-parallel acquisition function, parameter q was omitted.



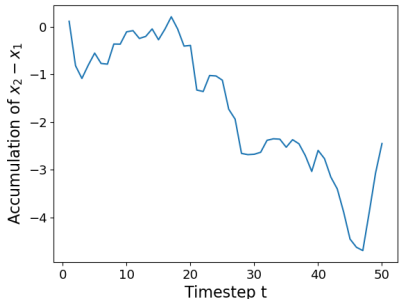
(a) Pareto optimality of CA-qEI



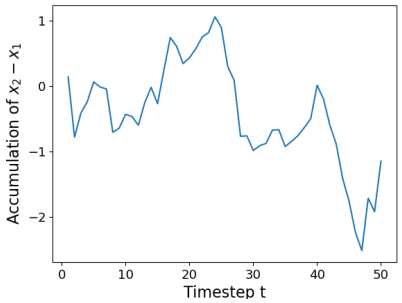
(b) Pareto optimality of qEI



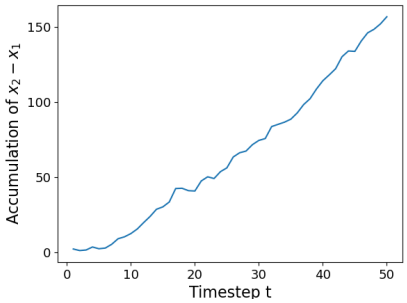
(c) Pareto optimality of CA-UCB



(d) Cost per dimension difference of CA-qEI

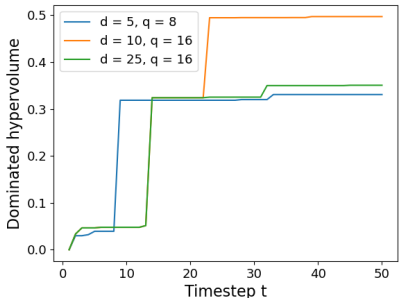


(e) Cost per dimension difference of qEI

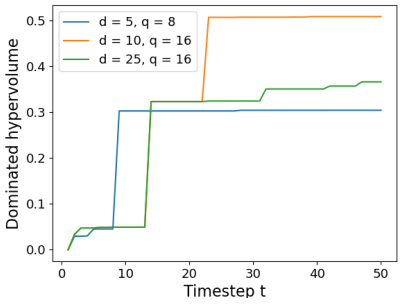


(f) Cost per dimension difference of CA-UCB

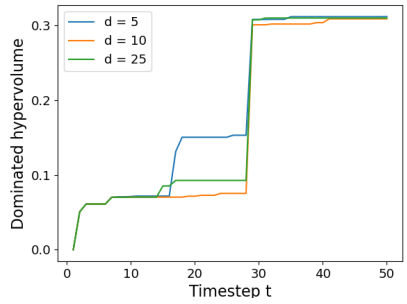
Figure 1: Multi-objective optimization of Cross-in-tray and Hölder table functions



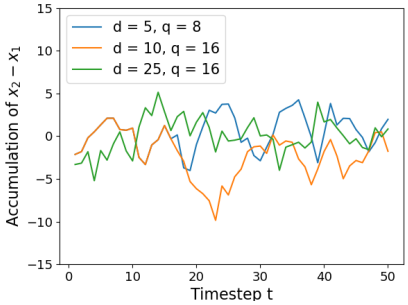
(a) Pareto optimality of CA-qEI



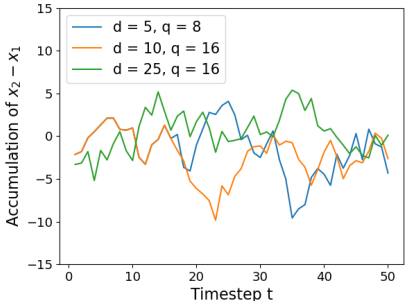
(b) Pareto optimality of qEI



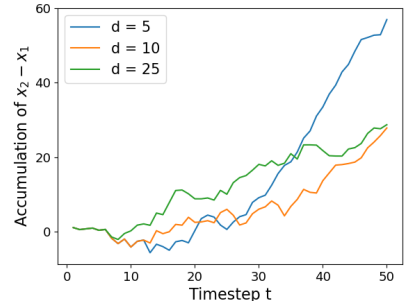
(c) Pareto optimality of CA-UCB



(d) Cost per dimension difference of CA-qEI



(e) Cost per dimension difference of qEI



(f) Cost per dimension difference of CA-UCB

Figure 2: Multi-objective optimization of Drop-wave and Rastrigin functions with delayed feedback

and that qEI is a noteworthy candidate for multi-objective optimization problems.

Secondly, the results of both experiments highlight that the heuristic proposed by Abdolshah et al. [4] revised for parallel acquisition functions (11) did not lead to cost-awareness in context of qEI. This indicates that qEI is cost-agnostic by nature, which might be attributed to the sensitivity of the acquisition function. In spite of a robust and representative sampling, only a small subset of inputs leads to an improvement. Consequently, the optimizer has little liberty in preserving the relative costs of the input dimensions. An alternative explanation could be that in order to maximize the expected improvement, qEI has to recommend a diverse batch of datapoints. However, this diversity averages out the aggregated sum of inputs and thus inhibits cost-awareness. Therefore, CA-qEI is unsuitable for settings wherein cost-awareness plays a major role.

Lastly, the second experiment has revealed that qEI retains its edge over UCB, despite having to perform serialized queries, thus bolstering the versatility of the former approach. However, tuning the batch size for a given delay is crucial, as the degree of success varies significantly based on the choice. The graphs indicate that a small or negative difference between the batch size and the delay might cause the model to adopt a judicious random search, given the similarity of the shape of the curves to random strategies. Meanwhile, CA-UCB seems to have relied on the cost-aware constraints for datapoint selection, as the model was unable to adjust the upper confidence bound while awaiting a response. This would explain the similar results across all three environment settings. Finally, inspection of a random strategy illustrates how harmful delayed environments can be, as only one configuration of qEI and CA-qEI managed to produce a noticeable improvement to the random method.

7 Responsible Research

The black-box functions used for evaluation are synthetic in origin and freely available, hence not posing any privacy concerns. To the best of our knowledge, findings described in this research paper do not create an opportunity for exploitation by malicious parties. The reported results can be reproduced by executing the main class of the provided codebase thanks to the fixed seed that eliminates any undesired randomness. The repository[2] also features a step-by-step installation guide and documentation in case that the reader would like to experiment with various hyperparameters. Furthermore, access to high-end hardware or a GPU is not a limiting factor, as the experiments are compatible even with less powerful devices.

8 Conclusions and Future Improvements

This research paper proposed an improvement to cost-aware optimization of multidimensional black-box functions. This improvement involved parallelization of the predictive model that outputs a batch of datapoints likely to yield an improvement.

[2]https://github.com/osihlovec/ca-qei

We have firstly investigated whether parallelization enhances optimality of multi-objective Bayesian otimizers. The experiments support this hypothesis, as parallel implementations managed to improve the optimality by approximately 16% in an environment with immediate feedback.

We have additionally sought to adapt the parallel expected gradient (qEI) method [7] in order to achieve cost-awareness in terms of multi-objective Bayesian optimization. Our results demonstrate that approach described by Abdolshah et al. [4] is unsuitable for qEI, as cost-aware parallel gradient (CA-qEI) has performed almost identically to its cost-agnostic counterpart. Contrasting these results to works shown to be cost-aware further reinforce this conclusion.

The third goal of this paper is to assess whether cost-aware parallelization diminishes the impact of environments with delayed responses. The second experiment has revealed that careful tuning of the degree of parallelism is crucial for alleviating the undesirable effect of such environments in terms of optimization. Finally, the results of this experiment indicate that model of [4] is still able to preserve cost-awareness, although the degree thereof is dependent on length of the delay.

An interesting avenue of future work would entail exploring other parallel acquisition functions. Based on the experiments, our method seems to only consider a very small selection of unexplored datapoints to be promising. A less discriminatory acquisition function could thus facilitate the cost-aware heuristic more effectively.

An additional point of improvement would be to perform a more thorough experimentation. The short time-frame of this research was a considerable limiting factor. As a result, there has been little room for increasing the number of timesteps, repetition of the experiments followed by construction of confidence intervals, or even an analysis of scalability with respect to higher input and output space dimensionalities. A rigorous evaluation of all aforementioned aspects would without a doubt augment the robustness of our research.

Finally, an alternative approach to cost-awareness is also worth entertaining. For example, rather than relying on a simplistic heuristic that conforms to the relative importance of input dimensions, one could employ Bayesian optimization to estimate the costs of unseen samples based on previous observations and attempt to minimize them as a secondary objective akin to works of [8] and [9].

References

[1] Apoorv Agnihotri and Nipun Batra. "Exploring Bayesian Optimization". In: *Distill* (2020). https://distill.pub/2020/bayesian-optimization. DOI: 10.23915/distill.00026.

[2] Vu Nguyen. "Bayesian Optimization for Accelerating Hyper-Parameter Tuning". In: *2019 IEEE Second International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*. 2019, pp. 302–305. DOI: 10.1109/AIKE.2019.00060.

[3] Min Jiang and Yimin Chen. "Research on bayesian optimization algorithm selection strategy". In: *The 2010 IEEE International Conference on Information and Automation*. 2010, pp. 2424–2427. DOI: 10.1109/ICINFA.2010.5512281.

[4] Majid Abdolshah et al. *Cost-aware Multi-objective Bayesian optimisation*. 2019. arXiv: 1909.03600 [cs.LG].

[5] Erich Merrill et al. "An Empirical Study of Bayesian Optimization: Acquisition Versus Partition". In: *Journal of Machine Learning Research* 22.4 (2021), pp. 1–25. URL: http://jmlr.org/papers/v22/18-220.html.

[6] Max Simchowitz et al. "Bayesian decision-making under misspecified priors with applications to meta-learning". In: *CoRR* abs/2107.01509 (2021). arXiv: 2107.01509. URL: https://arxiv.org/abs/2107.01509.

[7] Jialei Wang et al. *Parallel Bayesian Global Optimization of Expensive Functions*. 2019. arXiv: 1602.05149.

[8] Eric Hans Lee et al. *Cost-aware Bayesian Optimization*. 2020. arXiv: 2003.10870 [cs.LG].

[9] Gauthier Guinet, Valerio Perrone, and Cédric Archambeau. *Pareto-efficient Acquisition Functions for Cost-Aware Bayesian Optimization*. 2020. arXiv: 2011.11456 [cs.LG].

[10] Samuel Daulton, Maximilian Balandat, and Eytan Bakshy. *Parallel Bayesian Optimization of Multiple Noisy Objectives with Expected Hypervolume Improvement*. 2021. arXiv: 2105.08195 [cs.LG].

[11] Andreia P. Guerreiro, Carlos M. Fonseca, and Luı s Paquete. "The Hypervolume Indicator". In: *ACM Computing Surveys* 54.6 (July 2021), pp. 1–42. DOI: 10.1145/3453474. URL: https://doi.org/10.1145%2F3453474.

[12] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, 2006, pp. I–XVIII, 1–248. ISBN: 026218253X.

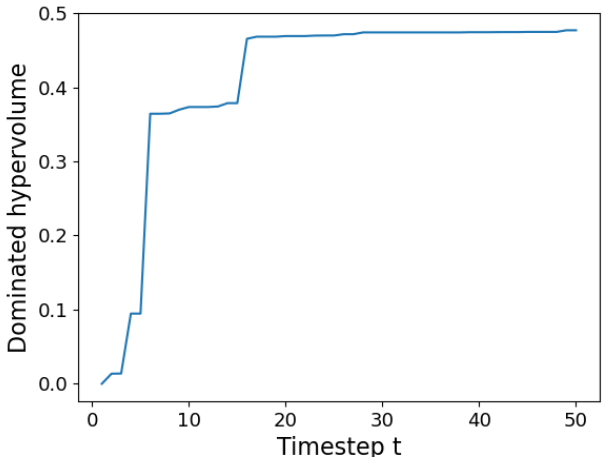
[13] Ali R. Al-Roomi. *Unconstrained Single-Objective Benchmark Functions Repository*. Halifax, Nova Scotia, Canada, 2015. URL: https://www.al-roomi.org/benchmarks/unconstrained.

[14] Momin Jamil, Xin-She Yang, and Hans-Jürgen Zepernick. "8 - Test Functions for Global Optimization: A Comprehensive Survey". In: *Swarm Intelligence and Bio-Inspired Computation*. Ed. by Xin-She Yang et al. Oxford: Elsevier, 2013, pp. 193–222. ISBN: 978-0-12-405163-8. DOI: https://doi.org/10.1016/B978-0-12-405163-8.00008-9. URL: https://www.sciencedirect.com/science/article/pii/B9780124051638000089.

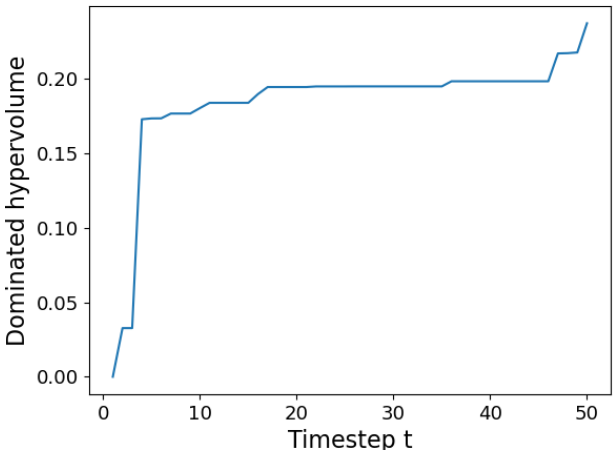
[15] Aimo. Törn and A. Zhilinskas. *Global optimization / Aimo Törn, Antanas Žilinskas*. English. Springer-Verlag Berlin ; New York, 1989, x, 255 p. : ISBN: 0387508716 3540508716. URL: http://www.loc.gov/catdir/enhancements/fy0814/89138974-t.html.

A Additional experiments

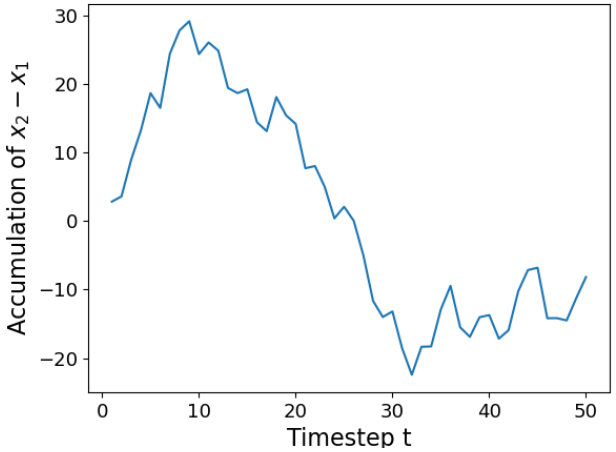
Figures 3a up to 3d demonstrate the performance of a random strategy in previously defined experiments. More specifically, this strategy involved uniform and independent sampling of 50 datapoints in an interval of $[0, 1]$ for both dimensions and then rescaling these inputs according to the search space of the given problem. Since the seed was fixed across all experiments, the samples in both experiments are equivalent, but have been rescaled differently. This explains the identical shape of cost per dimension in both experiments. The final difference is nonzero, however, the prior oscillation indicates that this is a temporary deviation, which would be corrected with additional timesteps.



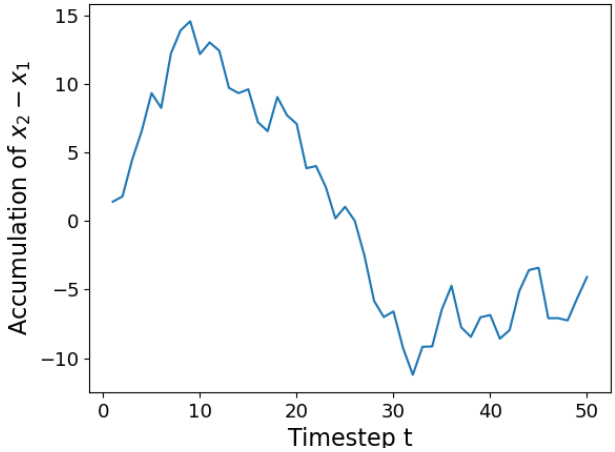
(a) Pareto optimality of a random strategy in a correct environment



(b) Pareto optimality of a random strategy in a delayed environment



(c) Cost per dimension difference of a random strategy in a correct environment



(d) Cost per dimension difference of a random strategy in a delayed environment

Figure 3: Evaluation of the performance a random strategy in previous experiments