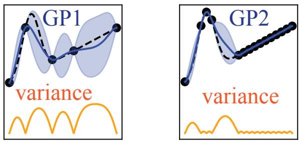
Adaptive Batch Sizes for Active Learning: A Probabilistic Numerics Approach

Masaki Adachi[1,3] **Satoshi Hayakawa**[2] **Martin Jørgensen**[4] **Xingchen Wan**[1]
Vu Nguyen[5] **Harald Oberhauser**[2] **Michael A. Osborne**[1]
[1]Machine Learning Research Group, University of Oxford
[2]Mathematical Institute, University of Oxford
[3]Toyota Motor Corporation
[4]Department of Computer Science, University of Helsinki
[5]Amazon

Abstract

Active learning parallelization is widely used, but typically relies on fixing the batch size throughout experimentation. This fixed approach is inefficient because of a dynamic trade-off between cost and speed—larger batches are more costly, smaller batches lead to slower wall-clock run-times—and the trade-off may change over the run (larger batches are often preferable earlier). To address this trade-off, we propose a novel Probabilistic Numerics framework that adaptively changes batch sizes. By framing batch selection as a quadrature task, our integration-error-aware algorithm facilitates the automatic tuning of batch sizes to meet predefined quadrature precision objectives, akin to how typical optimizers terminate based on convergence thresholds. This approach obviates the necessity for exhaustive searches across all potential batch sizes. We also extend this to scenarios with constrained active learning and constrained optimization, interpreting constraint violations as reductions in the precision requirement, to subsequently adapt batch construction. Through extensive experiments, we demonstrate that our approach significantly enhances learning efficiency and flexibility in diverse Bayesian batch active learning and Bayesian optimization applications.



batch size	Worst-case error (precision ε ≤ 1e-3)	
n=2	1e-1	1e-3
n=3	1e-2	1e-4
n=4	1e-3	1e-5
n=5	1e-4	1e-6

Figure 1: We fix the quadrature precision instead of batch size. The batch size changes adaptively to meet the predefined precision requirement. Our method, AdaBatAL, efficiently determines the optimal number of batch sizes and their querying positions without requiring a brute-force search of all possible batch sizes. AdaBatAL also offers adaptive batch sizes for constrained active learning and constrained Bayesian optimization.

1 Introduction

Active Learning (AL) (Settles, 2009) is a machine learning concept where the algorithm selects its training data, which enhances accuracy based on fewer labels. Its use is widespread in deep learning models (Gal et al., 2017; Ren et al., 2021; Kirsch et al., 2019) and Gaussian processes (GPs) (Houlsby et al., 2011; Riis et al., 2022). Bayesian AL intertwines with Probabilistic Numerics (PN) (Hennig et al., 2015, 2022), that reinterprets numerical tasks as Bayesian machine learning. This allows uncertainty to interlink with real-world constraints, improving empirical performance, and algorithmic flexibility. In PN, AL enables sample-efficient procedures, with Bayesian optimization (BO) and Bayesian quadrature (BQ) being key instances,

Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238. Copyright 2024 by the author(s).

applied in fields like drug discovery (Gómez-Bombarelli et al., 2018), materials (Adachi, 2021), and hyperparameter tuning (Feurer et al., 2015; Wu et al., 2020).

AL research can be classified into sequential and batch settings. While the sequential setting selects the next training data point one by one, the batch setting selects multiple points at the same time. We have two key metrics of performance: the number of iterations and the number of total queries. The number of iterations corresponds to the speed of model training, and the batch setting is advantageous as it can gain more feedback per iteration. In contrast, the number of total queries corresponds to the cost. For instance, labeling the data may involve expensive human evaluations. This total query metric is advantageous to the sequential setting as it can observe feedback for every single query to give a rational decision, whereas the batch setting needs to select multiple points without feedback.

However, situations arise where a balance between speed and cost is desirable. For instance, while training a model, renting cloud servers is an option, with charges applied based on the number of nodes (batch size) *and* duration (total queries). Another scenario is crowdsourcing annotation, where a balance is needed between the number of annotators (batch size) and the total working time (total queries). We aim to expedite model training while also saving on cost.

In addition to these situations, constraints often come into play in real-world applications, and often the constraints are also unknown a priori. Unknown constraints (Gelbart et al., 2014; Hernández-Lobato et al., 2016) are the constraints with which we must comply, but we do not know the constraint function a priori and are only observable pointwise. Hence, we had to estimate the true constraint function based on limited observations, resulting in uncertainty in the constraint estimation. For example, drug discovery needs to satisfy the safety constraints via animal experiments (Lipinski et al., 1997), but we do not know the functional form. Similarly, active learning with real physical experiments contains unknown constraints such as limitations from experimental apparatus or phase transition of measuring materials (Khatamsaz et al., 2023; Lookman et al., 2019). Training models on the cloud server may be halted due to errors or memory overflow, or annotators may pause annotation in cases of ambiguity in annotation guidelines or unclear samples. Avoiding querying samples that are likely to violate such unknown constraints is essential for the smooth execution of active learning. However, research on active learning under constraints is scarce, and no existing work considers adaptive batch size under constraints.

To address the said challenges, we propose a PN framework that adaptively adjusts the batch size. Figure 1 illustrates the concept. We hypothesize that an adaptive batch size can balance the trade-off between cost and speed. Fixed batch sizes might be ineffective because, as the shape of the acquisition function changes dynamically, the effectiveness of batch acquisition also shifts. In Figure 1, the left side displays four distinct peaks, indicating that four batch sizes would be suitable. Conversely, the right side exhibits only two prominent peaks, suggesting that two batch sizes would be more appropriate.

Given this intuition, we define batch construction as an approximation of a continuous target distribution (e.g., an acquisition function) using a discrete distribution (batch samples)—a process known as *quantization* applied in diverse machine learning fields (Graf & Luschgy, 2007; Karvonen, 2019; Teymur et al., 2021). The error in this approximation can be measured by the divergence between the target and the 'quantized' distribution. With this perspective, instead of fixing the batch size, we propose to fix the *precision* of the approximation. We reframe batch construction as a quantization task, assessing precision through divergence. We fix the precision requirement for iterations, allowing the batch size and locations to be adaptively adjusted. In essence, our approach quantifies numerical errors stemming from an insufficient batch sizes, strategically harnesses this computational uncertainty for decision-making, and embodies the essence of PN principles. Specifically, for GP models, this quantization links seamlessly to kernel quadrature (KQ), enabling the use of advanced KQ methods for efficient solutions. As such, we further re-cast the quantization task as a KQ task, using the worst-case integration error as our divergence metric. Our method, *adaptive batch active learning* (AdaBatAL), efficiently determines the optimal number of batch sizes and their querying positions without requiring a brute-force search of all possible batch sizes.

AdaBatAL also seamlessly handles AL in the presence of unknown constraints. We view the risk associated with these constraints as a 'varying precision requirement.' If querying points violate the constraints, we remove them from the valid dataset, thereby reducing precision. We interpret a high risk of constraint violation as a lower precision requirement and vice versa. Therefore, the constrained case serves as a 'preprocessing' step to determine the appropriate precision for AdaBatAL, with the constraint model estimating the precision requirement. The versatility of AdaBatAL provides a plug-and-play framework for AL, BQ, and BO, whether constraints are involved or not.

Contributions

1. **Adaptive batch size** We fixed *quadrature precision* via re-casting batch construction as a KQ, allowing batch size adaptively changing according to the acquisition function efficiently.
2. **Unknown constraints** We reinterpret the batch AL under unknown constraints as varying precision requirement. This allows adaptively changing the batch size and locations in accordance with the risks of constraint violation.
3. **Generality** Our adaptive batch construction scheme applies to AL, BO, and BQ by changing the target distribution of quantization with KQ. Moreover, it applies to non-continuous domains (e.g. combinatorial, mixed feature spaces).
4. **Significant improvement** is shown in both batch AL and batch BO tasks, outperforming 17 baselines over 6 synthetic and 7 real-world tasks.
5. **Open-source** we open-source the software on GitHub `https://github.com/ma921/AdaBatAL`.

2 Background

We start by providing the background on quantization and KQ. We then demonstrate the connection between GP, KQ, and BQ, leading to pure batch uncertainty sampling. We defer the background of GP and fully Bayesian GP (FBGP) in Supplementary B.1.

Quantization. Let μ be a probability distribution defined on a set $\mathcal{X}$. The *quantization* task is to find the discrete distribution $\nu := \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i}$, which best approximates μ with n representative points x_i. Here, δ_x denotes a point mass (delta distribution) located at $x \in \mathcal{X}$. To solve the quantization task, one first identifies an optimality criterion, typically a notion of *discrepancy* between μ and ν, and then develops an algorithm to approximately minimize it.

Kernel Quadrature. KQ is a numerical integration for calculating the integral of a function belonging to a reproducing kernel Hilbert space (RKHS). The aim is to find a good approximation of an, otherwise intractable, integral with a weighted sum. A KQ rule, $Q_{\boldsymbol{w},\boldsymbol{x}}$, is given by weights $\boldsymbol{w} = \{w_i\}_{i=1}^{n}$ and points $\boldsymbol{x} = \{x_i\}_{i=1}^{n}$,

$$Q_{\boldsymbol{w},\boldsymbol{x}}(f) := \sum_{i=1}^{n} w_i f(x_i) \approx \int_{\mathcal{X}} f(x)\mathrm{d}\mu(x), \qquad (1)$$

where f is a function of RKHS $\mathcal{H}$ associated with the kernel K. The *worst-case error* given μ and $\mathcal{H}$ is

$$\mathrm{wce}(Q_{\boldsymbol{w},\boldsymbol{x}}) := \sup_{\|f\|_{\mathcal{H} \leq 1}} \left| Q_{\boldsymbol{w},\boldsymbol{x}}(f) - \int_{\mathcal{X}} f(x)\mathrm{d}\mu(x) \right|. \quad (2)$$

The aim is to find $Q_{\boldsymbol{w},\boldsymbol{x}}$ minimizing worst-case error.

Connection to quantization. When inspecting the KQ rule as integration against a discrete distribution $\nu := \sum_{i=1}^{n} w_i \delta_{x_i}$, namely, $Q_{\boldsymbol{w},\boldsymbol{x}}(f) = \int_{\mathcal{X}} f(x)\mathrm{d}\nu(x)$, the worst-case error can be viewed as the *divergence* between μ and ν. Indeed, there is a theoretical connection between KQ and quantization, as KQ is the *weighted* quantization under the maximum mean discrepancy (MMD) metric (Karvonen, 2019; Teymur et al., 2021). MMD is a widely used method to quantify the divergence between two distributions (Sriperumbudur et al., 2010; Muandet et al., 2017), defined as:

$$\mathrm{MMD}_{\mathcal{H}}(\nu, \mu) := \left\| \int K(\cdot, x)\mathrm{d}\nu(x) - \int K(\cdot, x)\mathrm{d}\mu(x) \right\|_{\mathcal{H}},$$

and we can rewrite as (Huszár & Duvenaud, 2012):

$$\mathrm{MMD}^2_{\mathcal{H}}(\nu, \mu) := \sup_{\|f\|_{\mathcal{H}}=1} \left| \int f(x)\mathrm{d}\nu(x) - \int f(x)\mathrm{d}\mu(x) \right|^2.$$

This squared formulation is the same with the worst-case error. Therefore, solving KQ is equivalent to finding the discrete distribution ν that best approximates μ with regard to MMD. Note that KQ is a weighted quantization, unlike in the previous section.

Connection to Gaussian Process. Assume a function f is modelled by GP, $f \sim \mathcal{GP}(m, C)$, with limited number of observed points, $\mathcal{D}_0 := \{\boldsymbol{x}_0, \boldsymbol{y}_0\}$, where $\boldsymbol{y}_0 = f_{\mathrm{true}}(\boldsymbol{x}_0) + \epsilon$ are the noisy observations. We wish to estimate the expectation of the function $\hat{Z} := \int_{\mathcal{X}} f(x)\mathrm{d}\mu(x)$. This setting is called Bayesian quadrature (BQ) (O'Hagan, 1991), one of the central methods of PN. The integral estimate are as follows:

$$\mathbb{E}_f[\hat{Z}] = \int m(x)\mathrm{d}\mu(x) = \boldsymbol{z}^\top \boldsymbol{K}^{-1}\boldsymbol{y}_0, \qquad (3)$$

$$\mathbb{V}\mathrm{ar}_f[\hat{Z}] = \int C(x, x')\mathrm{d}\mu(x)\mathrm{d}\mu(x') = z' - \boldsymbol{z}^\top \boldsymbol{K}^{-1}\boldsymbol{z}, \qquad (4)$$

where $\boldsymbol{z} := \int K(x, \boldsymbol{x}_0)\mathrm{d}\mu(x)$ and $z' := \int K(x, x')\mathrm{d}\mu(x)\mathrm{d}\mu(x')$ are kernel mean and variance, respectively (see details in Supplementary B.3)

Huszár & Duvenaud (2012) proved the worst-case error (Eq. (2)) equals to the variance in Eq. (4) if quadrature nodes are $\mathcal{D}_0$. BQ expectation in Eq. (3) is a weighted sum; $\boldsymbol{z}^\top \boldsymbol{K}^{-1}\boldsymbol{y}_0 = \sum_{i=1}^{n} w_{BQ,i} y_i$, where $w_{\mathrm{BQ,\ j}} := \sum_{i=1}^{n} \boldsymbol{z}_i^\top \boldsymbol{K}_{i,j}^{-1}$. We can further think these weights as a discrete distribition $\nu_{\mathrm{BQ}} := \sum_{i=1}^{n} w_{\mathrm{BQ},i}\delta_{x_i}$, then the variance of integral estimation becomes:

$$\mathbb{V}\mathrm{ar}_f[\hat{Z}] = \mathrm{MMD}^2(\mu, \nu_{\mathrm{BQ}}) = \inf_{\boldsymbol{w}} \mathrm{wce}(Q_{\boldsymbol{w},\boldsymbol{x}})^2. \qquad (5)$$

This shows that KQ and BQ are closely connected (see details in (Huszár & Duvenaud, 2012)). The variance of integral is the uncertainty of GP over μ, so quantizing this distribution using KQ can be understood as a 'pure batch exploration' of GP uncertainty. This idea was applied to batch BQ (Adachi et al., 2022).

In summary. A quantization task can be viewed as a KQ task. The selected batch samples minimize the divergence between the target distribution μ and the batch samples ν, with a given kernel K. When we use the GP predictive covariance $C(\cdot,\cdot)$ as the kernel K for the MMD, the KQ becomes the pure batch exploration of GP uncertainty while also minimizing the divergence from the target distribution. Hence, batch construction via solving KQ can offer a quantization of the target distribution combined with uncertainty sampling.

3 Adaptive Batch Active Learning

Now, we introduce our method, AdaBatAL. Any KQ method can be used, but we employ the recombination (Hayakawa et al., 2022) for flexibility. We extend this to adaptive batch size under unknown constraints.

3.1 Problem Setting of Batch Active Learning

Batch Active Learning Consider we have a limited number of a labelled dataset $\mathcal{D}_0 = \{\boldsymbol{x}_k, \boldsymbol{y}_k\}_{k=1}^m$, and the large number of unlabelled pool set $\mathcal{X}_N = \{\boldsymbol{x}_l\}_{l=1}^N$, where $N \gg m$, an oracle can provide labels $\mathcal{Y}_N = \{\boldsymbol{y}_m\}_{m=1}^N$ for the corresponding inputs. We sequentially query the batch samples $\mathcal{D}_t^n = \{\boldsymbol{x}_j, \boldsymbol{y}_j\}_{j=1}^n$ with n batch sizes at t-th iteration, resulting in the total labelled dataset $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \mathcal{D}_t^n = (\mathcal{X}_t, \mathcal{Y}_t)$, and repeat T times[1]. The batch AL task is to select the $\mathcal{D}_t$ to minimize the prediction error between true labels $\mathcal{Y}_N$ and the prediction conditioned on $\mathcal{X}_N$ at given budget T. Throughout this paper, we assume the model is an FBGP for AL and a normal GP for BO.

Following the works Pinsler et al. (2019); Adachi et al. (2023a), we can recast the batch AL and batch BO as a quantization task. The difference between AL and BO comes down to the target distributions μ: the candidate pool of unlabelled inputs for batch AL, and the probability of global optimum location for batch BO. How to recast these tasks to a quantization is not a primary focus of this paper, we defer the explanation of their attempts in Supplementary D.1 and E.2. Important takeaways from their works are that the quantization approach can outperform popular baselines, such as BALD (Houlsby et al., 2011) for batch AL, and hallucination Azimi et al. (2010) for batch BO. Yet, their approach only considers fixed batch size without constraints. We augment their approaches by adaptive batch sizes under unknown constraints.

Unknown Constraints Consider our labelling scheme is subject to the constraint $c(x) \geq 0$, where c is the constraint with which we must comply, otherwise the query x is eliminated from the labelled dataset $\mathcal{D}_t$ (e.g., a drug candidate that breaches a safety constraint will not be tested.). We further assume the constraints are unknown a priori and are only observable pointwise. Hence, probabilistic model estimates the function $\hat{c}(x)$ with its predictive uncertainty, providing the probability of constraint satisfaction $q(x)$ at given input x. Following Gelbart et al. (2014), we model the constraint by another GP (see Supplementary E.4).

3.2 Problem Setting of Kernel Quadrature

As a general situation, consider we are given a kernel K on $\mathcal{X}$ and an N-point samples $\mathbf{X}_{\text{cand}} \in \mathcal{X}^N$ associated with a nonnegative weight $\mathbf{w}_{\text{cand}}$ with $\mathbf{w}_{\text{cand}}^\top \mathbf{1} = 1$[2]. We denote this as $\mu(x) := \sum_{i=1}^N w_i \delta_{x_i}$ as a discrete distribution, or $(\mathbf{w}_{\text{cand}}, \mathbf{X}_{\text{cand}})$ as the ordered pair. In a typical batch AL setting, μ is the candidate pool of unlabelled inputs with equal weights. The goal is to find a weighted subset $(\mathbf{w}_{\text{batch}}, \mathbf{X}_{\text{batch}})$, $\nu(x) := \sum_{j=1}^n w_j \delta_{x_j}$ which minimizes $\text{MMD}_{\mathcal{H}}(\mu, \nu)$ given μ and kernel K[3]. Hence, this is a KQ task. The quantized subset ν, $\mathbf{X}_{\text{batch}} \subset \mathbf{X}_{\text{cand}}$, will give the batch samples for batch AL and batch BO. Unlike the existing setting (Hayakawa et al., 2022; Adachi et al., 2022, 2023a), we additionally work under the following conditions:

(a) The upper bound of batch size n is given but the actual batch size is adaptively changed to meet the precision under the given tolerance ϵ_{LP}.

(b) After we choose the batch querying points, $(\mathbf{w}_{\text{batch}}, \mathbf{X}_{\text{batch}})$, each point $x \in \mathbf{X}_{\text{batch}}$ is subject to the probabilistic constraint $q(x)$[4] (and violated w.p. $1 - q(x)$), where $q : \mathcal{X} \to [0, 1]$ is given as GP. We query the true constraint $c(x)$, then we obtain the feasible points and corresponding weights $(\tilde{\mathbf{w}}_{\text{batch}}, \tilde{\mathbf{X}}_{\text{batch}})$, where $\tilde{\mathbf{X}}_{\text{batch}} \subset \mathbf{X}_{\text{batch}}$[5]. We use the feasible points for the quadrature.

(c) Additionally, a reward function $g : \mathcal{X} \to \mathbb{R}$ is given as additional flexibility that incorporates the other desideratum (e.g. soft constraint), and we want to make the expected reward $\tilde{\mathbf{w}}_{\text{batch}}^\top g(\tilde{\mathbf{X}}_{\text{batch}})$ as big as possible while making the worst-case error $\text{wce}(Q_{\tilde{\mathbf{w}}_{\text{batch}}, \tilde{\mathbf{X}}_{\text{batch}}})$[6] as small as possible.

[1] To clarify, $D_0 \subseteq D_t$ but $D_0 \not\subset D_t^n$.

[2] $\mathbf{1}$ is $[1, \dots, 1]^N$, the vector of ones.

[3] We set the kernel K as the posterior predictive covariance $C(\cdot,\cdot)$ (recall the background section).

[4] The true constraint $c(x)$ is deterministic but $q(x)$ becomes probabilistic due to the predictive uncertainty.

[5] $\tilde{\mathbf{X}}_{\text{batch}} = \mathbf{Z}^\top \mathbf{X}_{\text{batch}}$, where $\mathbf{Z}$ is a vector of Bernoulli random variables with probabilities $q(\mathbf{X}_{\text{batch}})$

[6] For brevity, b is batch, c is cand, then $\text{wce}(Q_{\tilde{\mathbf{w}}_\text{b}, \tilde{\mathbf{X}}_\text{b}}) = \tilde{\mathbf{w}}_\text{b}^\top K(\tilde{\mathbf{X}}_\text{b}, \tilde{\mathbf{X}}_\text{b}) \tilde{\mathbf{w}}_\text{b} - 2\tilde{\mathbf{w}}_\text{b}^\top K(\tilde{\mathbf{X}}_\text{b}, \mathbf{X}_\text{c}) \mathbf{w}_\text{c} + \mathbf{w}_\text{c}^\top K(\mathbf{X}_\text{c}, \mathbf{X}_\text{c}) \mathbf{w}_\text{c}$.

3.3 Kernel Quadrature via Nyström Approximation

Although the Nyström method (Williams & Seeger, 2000; Drineas & Mahoney, 2005; Kumar et al., 2012) is primarily used for approximating a large Gram matrix by a low-rank matrix, it can also be used for directly approximating the kernel function itself. Given a set of M points $X_{\text{nys}} = \{x_i\}_{i=1}^{M} \subset \mathcal{X}$, the Nyström approximation of $K(x, y)$ is given by:

$$K(x, y) \approx K_0(x, y) := \sum_{i=1}^{n-1} \lambda_i^{-1} \varphi_i(x) \varphi_i(y), \tag{6}$$

where $\varphi_i(\cdot) := u_i^\top K(X_{\text{nys}}, \cdot)$ $(i = 1, \dots, n-1)$ are called *test functions*, chosen from a larger M dimensional space $\text{span}\{K(x_i, \cdot)\}_{i=1}^{M}$. The Eq. (6) holds if $\lambda_s > 0$. To compute Eq. (6), we perform the best rank-s approximation of the Gram matrix $K(X_{\text{nys}}, X_{\text{nys}}) = U \Lambda U^\top$, given by eigendecomposition, where $U = [u_1, \dots, u_M] \in \mathbb{R}^{M \times M}$ is a real orthogonal matrix and $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_M)$ with $\lambda_1 \geq \ldots \geq \lambda_M \geq 0$.

We can use the test functions for integration estimator $\hat{Z} = \int_{\mathcal{X}} f(x) \mathrm{d}\mu(x)$. When the spectral decay in eigenvalues is steep, the Nyström method can give a good approximation of the original kernel function with a small number of test functions. Let $\boldsymbol{\varphi} = \{\varphi_1, \dots, \varphi_{n-1}\}^\top$ be the vector of test functions that spans $\mathcal{H}_{K_0}$, the RKHS associated with the approximated kernel K_0, we assume we have additional knowledge of expectations, namely, $\int_{\mathcal{X}} \boldsymbol{\varphi}(x) \mathrm{d}\mu(x) = \mathbf{w}_{\text{cand}}^\top \boldsymbol{\varphi}(\mathbf{X}_{\text{cand}})$ is given. We can actually construct a convex quadrature $Q_n = (w_i, x_i)_{i=1}^{n}$:

$$\sum_{i=1}^{n-1} w_i \varphi_i(x_i) = \int_{\mathcal{X}} \boldsymbol{\varphi}(x) \mathrm{d}\mu(x) \approx \int_{\mathcal{X}} f(x) \mathrm{d}\mu(x). \tag{7}$$

Now, we can approximate the integral by $n-1$ test functions. Hence, Eq. (7) can be understood as $n-1$ *equality constraints* which w_i and x_i need to satisfy.

The benefit of this approximation is to incorporate the information of spectral decay of Gram matrix for faster convergence. If the target function f is smooth, the spectral decay is fast, then the small number of test functions can well represent the function, leading to batch-size efficient AL and BO (Hayakawa et al., 2022; Adachi et al., 2022).

3.4 Linear Programming Formulation

To solve the above problem, we introduce the following linear programming (LP) problem that aims to achieve both the reward maximization and the worst-case error minimization where possible, given by modifying the algorithm adopted in (Adachi et al., 2023a) $(n \geq 3)$:

$$\begin{aligned}
&\underset{\mathbf{w}}{\text{maximize}} \quad \mathbf{w}^\top \big[g(\mathbf{X}_{\text{cand}}) \odot q(\mathbf{X}_{\text{cand}}) \big], \\
&\text{subject to} \\
&\big| (\mathbf{w} - \mathbf{w}_{\text{cand}})^\top \varphi_j(\mathbf{X}_{\text{cand}}) \big| \leq \epsilon_{\text{LP}} \sqrt{\lambda_j / (n-2)}, \\
&\forall j : 1 \leq j \leq n-2, \\
&(\mathbf{w} - \mathbf{w}_{\text{cand}})^\top q(\mathbf{X}_{\text{cand}}) \geq 0, \\
&\mathbf{w}^\top \mathbf{1} = 1, \quad \mathbf{w} \geq \mathbf{0}, \quad |\mathbf{w}|_0 \leq n,
\end{aligned}$$

where $\epsilon_{\text{LP}} \geq 0$ is a *tolerance* parameter, which can be interpreted as the quadrature precision requirements (smaller is more accurate), and (λ_j, φ_j) are given by the Nyström approximation (see 3.3)[7].

The intuition of this formulation is as follows:

(1) The solutions are the sparse weights $\mathbf{w}$, where the non-zero element of $\mathbf{w}$ corresponds to the batch selection, and the corresponding samples of $\mathbf{X}_{\text{cand}}$ is the batch samples $\mathbf{X}_{\text{batch}}$. We refer to the nonzero weights and corresponding samples as the solution $(\mathbf{w}_{\text{batch}}, \mathbf{X}_{\text{batch}})$[8], and its batch size is $|\mathbf{X}_{\text{batch}}| \leq n$. As such, this LP problem is to subsample the batch samples ν from the given discrete distribution μ, namely, quantization.

(2) The objective is to maximize the expected reward g under the risk of constraint violation q. This promotes safe sampling by increasing the expected constraints satisfaction $\mathbf{w}^\top q(\mathbf{X}_{\text{cand}})$.

(3) The first constraints correspond to equality constraints with test functions in Eq. (7). We relaxed the equality constraints to inequality constraints to accept the tolerance ϵ_{LP}. These $n-2$ inequality constraints restrict the solution space to where the approximation error of the expectations of test functions $|(\mathbf{w} - \mathbf{w}_{\text{cand}})^\top \varphi_j(\mathbf{X}_{\text{cand}})|$[9] is within the tolerance parameter ϵ_{LP}. These $n-2$ constraints are very restrictive; the flexibility to select the larger objective is much more restricted than the typical LP problem. ϵ_{LP} controls the trade-off between the accuracy for quadrature and relaxing solution space to find the larger objective.

(4) Other constraints assure the number of nonzero elements of the solution set $\mathbf{w}$ is fewer than the upper bound of batch size n, the convex and positive weights, and the probability of probabilistic constraints' satisfaction is positive.

Thus, in response to conditions (b)(c), the solution of this LP problem provides the batch samples that satisfy convex quadrature rules within the tolerance *and* maximizing the reward. The balance between

[7] $\odot$ refers to Hadamard product, and $|\cdot|_0$ denotes the number of nonzero entries.

[8] $\mathbf{X}_{\text{batch}} \subset \mathbf{X}_{\text{cand}}$, $\mathbf{w}_{\text{batch}} \subset \mathbf{w}_{\text{cand}}$, and $|\mathbf{X}_{\text{batch}}| = |\mathbf{w}|_0$

[9] This is a quadrature error in Eq. (7). $\big|\mathbf{w}^\top \boldsymbol{\varphi}(\mathbf{X}_{\text{cand}}) - \mathbf{w}_{\text{cand}}^\top \boldsymbol{\varphi}(\mathbf{X}_{\text{cand}})\big| \approx \big|\int_{\mathcal{X}} f(x) \mathrm{d}\nu(x) - \int_{\mathcal{X}} f(x) \mathrm{d}\mu(x)\big|$.

quadrature accuracy and reward maximization can be controlled by a single parameter ϵ_{LP}. To be clear, only within §3.4, the term 'constraints' refers to the ones in LP formulations. Otherwise, the constraints refer to the task-specific unknown constraints (e.g. safety constraints for drug discovery).

3.5 Adaptive Batch Sizes

The count of non-zero elements, denoted as $|\boldsymbol{w}|_0$, is adjusted based on the tolerance ϵ_{LP}. The intuition of the batch size adaptivity is explicated as:

1. Higher precision demands result in a smaller quadrature error tolerance. This necessitates a larger sample set for more precise integration.
2. Conversely, lower precision requirements needs fewer $|\boldsymbol{w}|_0$ to meet the desired accuracy.

Elaborating further, the batch size is tied to slack variables in LP solvers. As the tolerance ϵ_{LP} increases, some inequality constraints become deactivated (Dantzig, 2002). The batch size is determined by the number of active constraints, often leading to sparse weights with $|\boldsymbol{w}|_0 < n$. When constraints are loose, a large preset batch size is inefficient, as the desired precision can be achieved with fewer samples. As such, we can identify the adaptive batch size $|\boldsymbol{w}|_0$ without needing a brute-force search of all possible batch sizes.

Note that ϵ_{LP} controls *all* balances: the batch size, quadrature accuracy, and reward maximization. Interestingly, its behavior is not a monotonic decrease in its magnitude. As ϵ_{LP} approaches infinity, the batch size converges to 1, aligning with the sequential AL case. An increased ϵ_{LP} shrinks the batch size as observed in §5.1. This approach is essentially a heuristic for adaptive batch sizes. Although it satisfies a predefined worst-case error threshold, it does not guarantee optimal results based on other established metrics like mutual information (Krause & Guestrin, 2012). However, as Leskovec et al. (2007) highlighted, when greedily maximizing mutual information under the weighted candidates and a budget constraint (limitation in the number of the total queries), the approximation factor can be arbitrarily bad. Hence, even popular strategies, such as BALD (Houlsby et al., 2011), also cannot achieve a solution within $1 - 1/e$ of the optimal in our problem setting (Li et al., 2022).

3.6 Unknown Constraints As The Lowered Precision Requirement

In this further examination, we address the probabilistic constraint denoted as q. Given the uncertainty in predicting the true constraint c, the candidate solution, $\mathbf{X}_{\text{cand}}$, carries a risk of violation. We can estimate the *expected* violation rate by $\epsilon_{\text{vio}} := 1 - \mathbf{w}_{\text{cand}}^{\top} q(\mathbf{X}_{\text{cand}})$. It is assumed that infeasible points are eliminated from quadrature nodes for computation, reducing quadrature accuracy. The expected violation rate ϵ_{vio} can be interpreted as the *risk* we cannot control. A high-risk scenario necessitates cautious exploration to avoid wasting valuable queries; this suggests smaller batch sizes and selecting queries where $\mathbf{x}_{\text{cand}}$ is more likely to satisfy the true constraint c. Conversely, a low risk allows for more optimistic exploration.

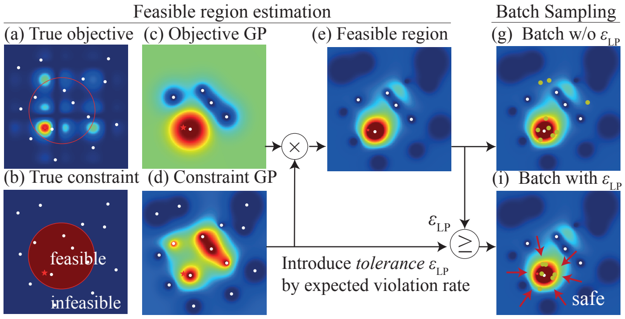

Feasible region estimation
Batch Sampling
(a) True objective
(c) Objective GP
(e) Feasible region
(g) Batch w/o ε_{LP}
(b) True constraint
(d) Constraint GP
(i) Batch with ε_{LP}
feasible
infeasible
$\times$
ε_{LP}
$\geq$
Introduce *tolerance* ε_{LP} by expected violation rate
safe

Figure 2: Constrained batch active learning. As the increased violation risk ϵ_{vio} propagates to the tolerance ϵ_{LP}, reward maximization is subsequently prioritized over quadrature, resulting in safe batch samples.

In response to varying risk levels, we advocate for an *adaptive* exploration strategy. Our proposed method is straightforward yet effective: setting $\epsilon_{\text{LP}} = \epsilon_{\text{vio}}$. This approach allows for automatic adjustment of exploration safety. When ϵ_{vio} is high, indicating greater risk, ϵ_{LP} is set higher. This results in looser quadrature precision, smaller batch sizes, and a solution space that is more likely to satisfy constraints[10]. Thus, a higher ϵLP leads to safer batch sampling. When the risk ϵ_{vio} is low, ϵ_{LP} is set lower, allowing for larger batch sizes and more explorative solutions. Figure 2 demonstrates this adaptive behavior: high-risk information ϵ_{vio} influences ϵ_{LP}, leading to safer batch samples. Adaptive safe exploration is not necessarily always safe. We need to explore uncertain regions at some point and we propose it is when the risk is low. Our PN framework effectively bridges computational uncertainty and real-world risk, providing an automated and adaptable balance between safety and exploration.

[10] Remember that a reduction in quadrature precision results in an expansion of the solution space, which in turn enables the identification of solutions with higher LP objective values, as denoted by $\mathbf{w}_{\text{cand}}^{\top}\left[g(\mathbf{X}_{\text{cand}}) \odot q(\mathbf{X}_{\text{cand}})\right]$, where $\mathbf{w}_{\text{cand}}^{\top} q(\mathbf{X}_{\text{cand}})$ represents the expected satisfaction of the constraint. Thus, maximizing the LP objective value leads to increasing the constraint satisfaction probability.

3.7 Error Bounds

The error estimate of KQ is essentially determined by the approximation error of the Nyström method, $\epsilon_{\text{nys}} := \max_{x \in \mathbf{X}_{\text{cand}}} |K_0(x,x) - K(x,x)|^{1/2}$. Error bounds for this approximation have been well studied in the literature (Drineas & Mahoney, 2005; Kumar et al., 2012; Hayakawa et al., 2023).

Proposition 1. *Under the above setting, let $\mathbf{w}_*$ be the optimal solution of the LP, and let $\mathbf{X}_{\text{batch}}$ be the subset of $\mathbf{X}_{\text{cand}}$, corresponding to the nonzero entries of $\mathbf{w}_*$ (denoted by $\mathbf{w}_{\text{batch}}$). Suppose that $\tilde{\mathbf{X}}_{\text{batch}}$ is given by a random subset of $\mathbf{X}_{\text{batch}}$, where each point x satisfies the constraints with probability $q(x)$, and let $\tilde{\mathbf{w}}_{\text{batch}}$ be the corresponding weights. Then, we have*

$$\mathbb{E}[\tilde{\mathbf{w}}_{\text{batch}}^\top g(\tilde{\mathbf{X}}_{\text{batch}})] \geq \mathbf{w}_{\text{cand}}^\top \left[g(\mathbf{X}_{\text{cand}}) \odot q(\mathbf{X}_{\text{cand}})\right], \tag{8}$$

and, for any function f in the RKHS with kernel K,

$$\begin{aligned}&\mathbb{E}\left[\left|\tilde{\mathbf{w}}_{\text{batch}}^\top f(\tilde{\mathbf{X}}_{\text{batch}}) - \mathbf{w}_{\text{cand}}^\top f(\mathbf{X}_{\text{cand}})\right|\right] \\ &\leq (\epsilon_{\text{vio}} K_{\max} + 2\epsilon_{\text{nys}} + \epsilon_{\text{LP}})\|f\|,\end{aligned} \tag{9}$$

where $\|f\|$ is the RKHS norm of f, $K_{\max} := \max_{x \in \boldsymbol{X}_{cand}} K(x,x)^{1/2}$, and $\epsilon_{\text{vio}} := 1 - \mathbf{w}_{\text{cand}}^\top q(\mathbf{x}_{\text{cand}})$ is the expected violation rate with respect to the empirical measure given by $(\mathbf{w}_{\text{cand}}, \mathbf{X}_{\text{cand}})$.

The proof is given in Supplementary A. This proposition indicates that we can obtain a quantitative estimate of the two tasks described in (c) concurrently. We can attain at least the expected reward of the original batch while ensuring that the resulting measure (which may not necessarily be probabilistic) integrates the functions in the RKHS within a proven error.

3.8 How to Solve The LP Problem

We used Gurobi (Gurobi Optimization, LLC, 2024) to solve the LP problem. We used the randomized singular value decomposition to eigendecompose the Gram matrix (Halko et al., 2011) with M-point samples $\mathbf{X}_{\text{nys}} \subset \mathbf{X}_{\text{cand}}$. We set $\epsilon_{\text{LP}} = 10^{-8}$ as the lower bound to avoid LP failure due to the randomness of μ. The complexity of this computation is lower than $\mathcal{O}(NM + M^2 \log n + Mn^2 \log(N/n))$ (Hayakawa et al., 2022).

Probability function q A probability function q can be a given constraint function (Gardner et al., 2014), or estimated function as another GP (Gelbart et al., 2014) (see details in Supplementary E.4). If there is no constraints, we can simply set $q(x) = 1$, then it becomes standard batch AL, BQ, or BO.

Reward function g A reward function g is for an additional flexibility to incorporate the information. If we do not have particularly informative information to add, we can simply set as $g = 1$. We can view g as the soft constraint of the objective. We can set g for another acquisition function, or prior knowledge of global optimum such as Hvarfner et al. (2022); Adachi et al. (2024).

4 Related Work

Batch Active Learning and Optimization There are a wide variety of batch methods has been proposed: (1) batch AL; for kernels (Kremer et al., 2014; Joshi et al., 2009; Leskovec et al., 2007; Riis et al., 2022), deep learning (Gal et al., 2017; Kirsch et al., 2019; Sener & Savarese, 2018; Pinsler et al., 2019). (2) batch BQ (Wagstaff et al., 2018; Adachi et al., 2022, 2023b), (3) batch BO, a greedy extension of sequential algorithms (Azimi et al., 2010; González et al., 2016; Eriksson et al., 2019; Balandat et al., 2020), diversified batch with determinantal point process (DPP) (Kathuria et al., 2016; Nava et al., 2022). Constrained batch sampling has been researched in BO (Hernández-Lobato et al., 2016; Letham et al., 2019; Eriksson & Poloczek, 2021). However, most do not discuss the quality of batch construction, like KQ methods. The adaptive batch size setting only found in BO (Nguyen et al., 2016), to the best of our knowledge.

Kernel Quadrature There are a number of KQ algorithms; herding/optimization (Chen et al., 2010; Bach et al., 2012; Huszár & Duvenaud, 2012), random sampling (Bach, 2017; Belhadji et al., 2019), DPP (Belhadji et al., 2019; Belhadji, 2021), kernel thinning (Dwivedi & Mackey, 2021, 2022), recombination (Hayakawa et al., 2022, 2023), kernel Stein discrepancy (Chen et al., 2018, 2019; Teymur et al., 2021), randomly pivoted Cholesky (Epperly & Moreno, 2023). While any KQ algorithms can be used to solve our problems, we focused on the recombination algorithm due to its flexibility.

5 Experiments

We evaluate our new algorithm, *AdaBatAL*, on synthetic and real-world tasks on batch AL and BO, with and without probabilistic constraints. We implemented AdaBatAL using PyTorch (Paszke et al., 2019), GPyTorch (Gardner et al., 2018), BoTorch (Balandat et al., 2020), and SOBER (Adachi et al., 2023a). All experiments were averaged over 10 repeats, and performed on a laptop[11]. We fix the number of initial random samples for objective queries to $n_{\text{obj}} = 10$. The details on experimental conditions and background on real-world examples are summarized in Supplementary G.

[11] Performed on MacBook Pro 2019, 2.4 GHz 8-Core Intel Core i9, 64 GB 2667 MHz DDR4

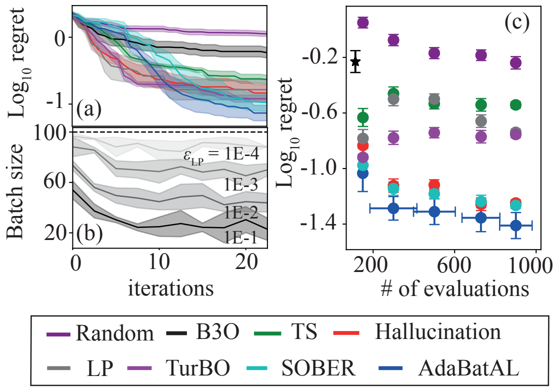


Figure 3: Batch Bayesian optimization results on Hartmann ($d = 6$): (a) convergence plot with ($n \leq 5$). (b) batch size variability ($n \leq 100$). The tolerance is set ($\epsilon = 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}$). (c) Total queries vs. simple regret at the last iteration results of (a)(b). For fixed batch size methods, the mean batch size of AdaBatAL is used ($n = 5, 30, 50, 73, 90$). The plot shows mean ± standard error of the mean.

5.1 Efficacy of Adaptive Batch Size

We first investigate the effect of the adaptive batch size itself *without* unknown constraints, namely, $q(x) = 1$. To compare with the only baseline of the adaptive batch size method, B3O (Nguyen et al., 2016), we selected the batch BO setting. We compared AdaBatAL with the 6 popular baselines of batch BO; B3O, Thompson sampling (TS) (Kandasamy et al., 2018), hallucination (Azimi et al., 2010), local penalization (LP)[12] (González et al., 2016), TurBO (Eriksson et al., 2019), SOBER (Adachi et al., 2023a).

Figure 3 illustrates that AdaBatAL consistently outperformed the baselines throughout the experiments. An increase in the tolerance ϵ_{LP} results in a reduced batch size. Over iterations, the batch size decreases for all values of ϵ_{LP}. This indicates that AdaBatAL initially needs more exploratory samples, then it squeezes its search space for exploitation. When matched against fixed batch size methods with a total cost, AdaBatAL achieves a lower regret for the same budget, even when compared to the original SOBER. While B3O tends to opt for a small batch size of around 4, AdaBatAL can adjust its batch size by ϵ_{LP}.

5.2 Efficacy of Expected Violation Rate

We empirically examine the role of expected violation rate ϵ_{vio} in constrained BO as the time-varying tolerance ϵ_{LP}. Figure 4 presents the main findings.

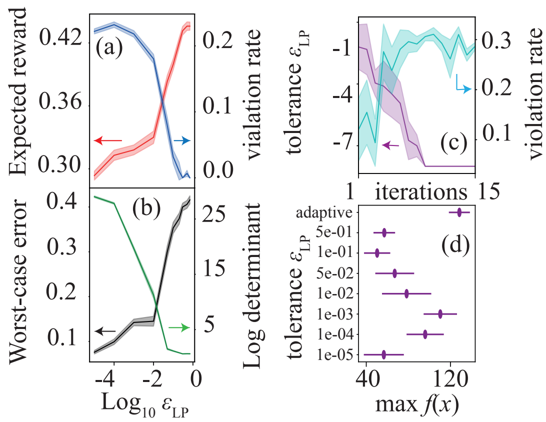


Figure 4: Tolerance effect on constrained batch BO on Branin ($d = 2$): the balance between (a) violation rate and expected reward, and (b) worst-case error and log determinant. (c) Tolerance adaptively controls violation rate, and (d) outperforms the fixed cases. (a)(b)(c) are the two Y-axis plots where the color and arrow indicate which Y axis to see.

Four key metrics

1. *The expected reward* (LP objective): the proxy for how safely we explore.
2. *The violation rate* $1 - |\tilde{\mathbf{X}}_{\text{batch}}|/|\mathbf{X}_{\text{batch}}|$: the proxy for actual results on how safely we explore.
3. *The worst-case error* $\text{wce}(Q_{\tilde{\mathbf{w}}_{\text{batch}}, \tilde{\mathbf{X}}_{\text{batch}}})$: the precision of quadrature.
4. *log determinant* $\log|K(\tilde{\mathbf{X}}_{\text{batch}}, \tilde{\mathbf{X}}_{\text{batch}})|$: the proxy for how diversely we explore.

We examined the impact of ϵ_{LP} on four key metrics, as discussed in § 3.6. We aligned ϵ_{LP} with ϵ_{vio} to facilitate *adaptive* exploration relative to the specified risk level ϵ_{vio}. The x-axis in Figures (a) and (b) represents variations in ϵ_{vio}. At higher risk levels, it is essential to prioritize safety. Consequently, there is an increase in the expected reward, correlating with a higher likelihood of constraint satisfaction. This relationship is evident through a reduction in the violation rate, which signifies safer exploration practices. The numerical metrics give insights of safety exploration in the numerical level: High tisk leads to an increase in the worst-case error, which reflects a relaxation in precision requirements. A smaller log determinant suggests less diversity in batch samples, indicated by the proximity of the selected points $\mathbf{X}_{\text{cand}}$ to each other. Conversely, at lower risk levels, we observe a trend towards more optimistic and exploratory sampling. Hence, our findings confirm that by setting $\epsilon_{\text{LP}} = \epsilon_{\text{vio}}$, our batch exploration successfully adapts to varying risk levels.

We further examined the evolution of the expected violation rate ϵ_{vio} during the optimization loop. As depicted in Figure 4 (c), the expected violation rate

[12] Only within this section 5.1, LP refers to local penalization. Otherwise, LP means linear programming.

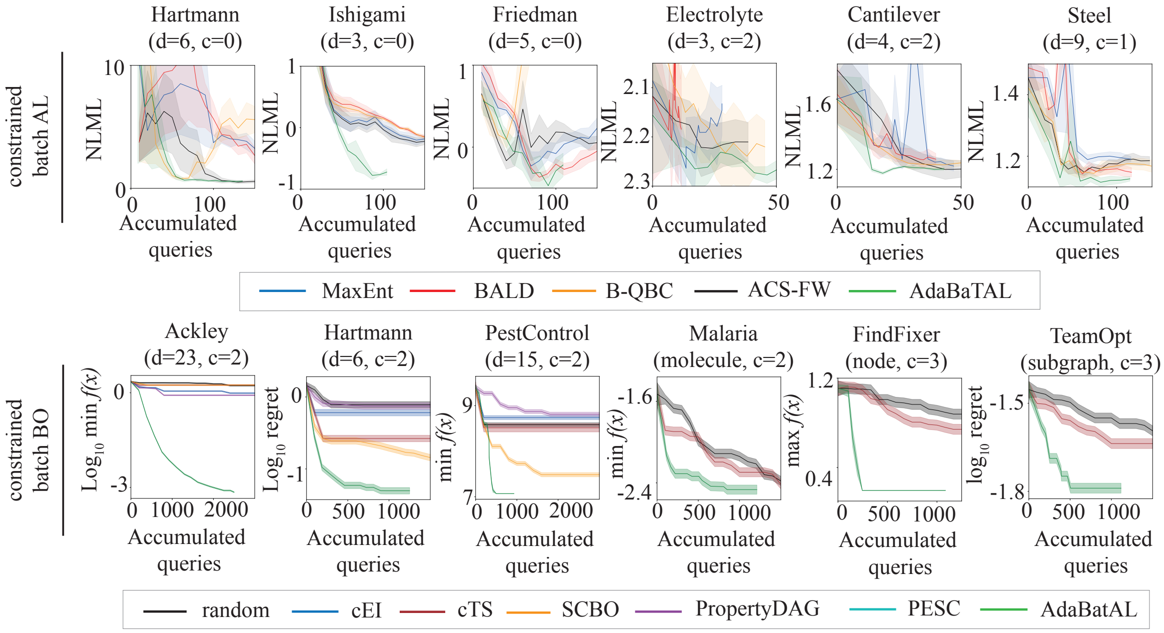


Figure 5: Convergence plot of both constrained batch active learning and Bayesian optimization results across 5 synthetic functions and 7 real-world tasks . d is the dimension, c is the number of unknown constraints. Negative log marginal likelihood (NLML) for active learning tasks, log regret or log best observations for optimization task. Lines and shaded area denote mean ± 1 standard error.

$\epsilon_{\text{vio}} = \epsilon_{\text{LP}}$ begins high and diminishes to a minimal value over time. This trend suggests an initial emphasis on safely gathering data, transitioning to greater exploration later on. This approach mirrors strategies like 'safe' BO (Sui et al., 2015), which has demonstrated strong empirical performance (e.g., Figure 4 in Xu et al. (2023)) backed by theoretical guarantee. The adaptive tolerance inherently exhibits this behavior with adaptive batch size. Moreover, Figure 4 (d) indicates that adaptive tolerance converges more rapidly than fixed versions. Notably, the most effective fixed tolerance was $\epsilon_{\text{LP}} = 10^{-3}$, suggesting that even in the absence of adaptive tolerance, AdaBatAL outperforms the original SOBER ($\epsilon_{\text{LP}} = 0$) under constraints.

5.3 Empirical Evaluation

We tested AdaBatAL's empirical performance across diverse tasks. For batch AL, we compared against five baselines: MaxEnt (MacKay, 1992), BALD (Houlsby et al., 2011; Kirsch et al., 2019), B-QBC (Riis et al., 2022), and ACS-FW (Pinsler et al., 2019). We evaluated on three synthetic and three real-world tasks. For batch BO, we also explored constrained batch BO and compared against five popular baselines: random, cEI (Letham et al., 2019), PESC (Hernández-Lobato et al., 2016), SCBO (Eriksson & Poloczek, 2021), and cTS (Eriksson & Poloczek, 2021). The Malaria, FindFixer, and TeamOpt tasks involve non-continuous inputs over non-Euclidean spaces, each requiring specialized kernels (Tanimoto kernel (Ralaivola et al., 2005) for molecules and the diffusion graph kernel (Zhi et al., 2023) for graphs). Due to this unique and crucial real-world setting, the only comparable baselines were random and cTS. Others utilized a standard GP with an RBF kernel. It is important to note that this is *constrained* batch BO, which differs from normal batch BO. Typically, constrained batch BO extends the standard acquisition function with regular batch methods. We chose cEI and cTS as representative methods for these approaches. More details are available in Supplementary G. Figure 5 shows AdaBatAL's strong empirical performance.

6 Discussion and Limitations

We introduced AdaBatAL, a versatile approach capable of adaptive batch sizes under probabilistic constraints for both AL and BO. It is also applicable for non-continuous inputs (e.g., strings for drug discovery and graphs for social data) and arbitrary acquisition functions as the reward function. AdaBatAL is best suited for batch sizes larger than three and does not support asynchronous batch settings (Kandasamy et al., 2018). Its efficacy in high-dimensional BO, which often faces challenges with slow eigenvalue decay, remains an open problem. However, the error bounds of the Nyström method are not directly related to dimensionality; rapid convergence is possible if the function exhibits fast eigenvalue decay, as in the case of the Ackley function.

Acknowledgements

We thank Samuel Daulton, Siu Lun Chau, Wenjie Xu, and Pierre Osselin for helpful feedback on our paper, and anonymous reviewers who gave useful comments. Masaki Adachi was supported by the Clarendon Fund, the Oxford Kobe Scholarship, the Watanabe Foundation, and Toyota Motor Corporation. Satoshi Hayakawa was supported by the Clarendon Fund, the Oxford Kobe Scholarship, and the Toyota Riken Overseas Scholarship. Harald Oberhauser was supported by the DataSig Program [EP/S026347/1], the Hong Kong Innovation and Technology Commission (InnoHK Project CIMDA), and the Oxford-Man Institute. Martin Jørgensen was partly supported by the Research Council of Finland (grant 356498).

References

Masaki Adachi. High-dimensional discrete Bayesian optimization with self-supervised representation learning for data-efficient materials exploration. In *NeurIPS 2021 AI for Science Workshop*, 2021. doi: https://openreview.net/forum?id=xJhjehqjQeB.

Masaki Adachi, Satoshi Hayakawa, Martin Jørgensen, Harald Oberhauser, and Michael A Osborne. Fast Bayesian inference with batch Bayesian quadrature via kernel recombination. *Advances in Neural Information Processing Systems*, 35, 2022. doi: https://doi.org/10.48550/arXiv.2206.04734.

Masaki Adachi, Satoshi Hayakawa, Saad Hamid, Martin Jørgensen, Harald Oberhauser, and Michael A Osborne. SOBER: Highly parallel Bayesian optimization and Bayesian quadrature over discrete and mixed spaces. In *ICML 2023 Workshop: Sampling and Optimization in Discrete Space*, 2023a. doi: https://doi.org/10.48550/arXiv.2301.11832.

Masaki Adachi, Yannick Kuhn, Birger Horstmann, Arnulf Latz, Michael A Osborne, and David A Howey. Bayesian model selection of lithium-ion battery models via Bayesian quadrature. *IFAC-PapersOnLine*, 56(2):10521–10526, 2023b. doi: https://doi.org/10.1016/j.ifacol.2023.10.1073.

Masaki Adachi, Brady Planden, David A Howey, Krikamol Muandet, Michael A Osborne, and Siu Lun Chau. Looping in the human: Collaborative and explainable Bayesian optimization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2024. doi: https://doi.org/10.48550/arXiv.2310.17273.

Raul Astudillo and Peter Frazier. Bayesian optimization of function networks. *Advances in neural information processing systems*, 34:14463–14475, 2021.

Javad Azimi, Alan Fern, and Xiaoli Fern. Batch Bayesian optimization via simulation matching. *Advances in Neural Information Processing Systems*, 23, 2010.

F. Bach. On the equivalence between kernel quadrature rules and random feature expansions. *Journal of Machine Learning Research*, 18:714, 2017.

F. Bach, S. Lacoste-Julien, and G. Obozinski. On the equivalence between herding and conditional gradient algorithms. In *International Conference on Machine Learning (ICML)*, 2012.

Maximilian Balandat, Brian Karrer, Daniel Jiang, Samuel Daulton, Ben Letham, Andrew G Wilson, and Eytan Bakshy. BoTorch: a framework for efficient Monte-Carlo Bayesian optimization. *Advances in neural information processing systems*, 33:21524–21538, 2020.

Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *science*, 286(5439): 509–512, 1999.

A. Belhadji, R. Bardenet, and P. Chainais. Kernel quadrature with DPPs. In *International Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

Ayoub Belhadji. An analysis of ermakov-zolotukhin quadrature using kernels. *Advances in Neural Information Processing Systems*, 34:27278–27289, 2021.

Eli Bingham, Jonathan P Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D Goodman. Pyro: Deep universal probabilistic programming. *The Journal of Machine Learning Research*, 20(1):973–978, 2019.

François-Xavier Briol, Chris J Oates, Mark Girolami, Michael A Osborne, and Dino Sejdinovic. Probabilistic integration: A role in statistical computation? *Statistical Science*, 34(1):1–22, 2019.

Mark S Butler. The role of natural product chemistry in drug discovery. *Journal of natural products*, 67 (12):2141–2153, 2004.

Trevor Campbell and Tamara Broderick. Automated scalable Bayesian inference via Hilbert coresets. *The Journal of Machine Learning Research*, 20(1):551–588, 2019.

Jerry F Casteel and Edward S Amis. Specific conductance of concentrated solutions of magnesium salts in water-ethanol system. *Journal of Chemical and Engineering Data*, 17(1):55–59, 1972.

Naitong Chen, Zuheng Xu, and Trevor Campbell. Bayesian inference via sparse Hamiltonian flows. *Advances in Neural Information Processing Systems*, 35: 20876–20888, 2022.

Wilson Ye Chen, Lester Mackey, Jackson Gorham, François-Xavier Briol, and Chris Oates. Stein points. In *International Conference on Machine Learning*, pp. 844–853. PMLR, 2018.

Wilson Ye Chen, Alessandro Barp, François-Xavier Briol, Jackson Gorham, Mark Girolami, Lester Mackey, and Chris Oates. Stein point Markov chain Monte Carlo. In *International Conference on Machine Learning*, pp. 1011–1021. PMLR, 2019.

Y. Chen, M. Welling, and A. Smola. Super-samples from kernel herding. In *International Conference on Uncertainty in Artificial Intelligence (UAI)*, 2010.

George B Dantzig. Linear programming. *Operations research*, 50(1):42–47, 2002.

Samuel Daulton, Maximilian Balandat, and Eytan Bakshy. Differentiable expected hypervolume improvement for parallel multi-objective Bayesian optimization. *Advances in Neural Information Processing Systems*, 33:9851–9864, 2020.

Samuel Daulton, Maximilian Balandat, and Eytan Bakshy. Parallel Bayesian optimization of multiple noisy objectives with expected hypervolume improvement. *Advances in Neural Information Processing Systems*, 34:2187–2200, 2021.

Petros Drineas and Michael W. Mahoney. On the Nyström method for approximating a Gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, 6(72):2153–2175, 2005.

Raaz Dwivedi and Lester Mackey. Kernel thinning. In *Proceedings of Thirty Fourth Conference on Learning Theory*, volume 134 of *Proceedings of Machine Learning Research*, pp. 1753–1753. PMLR, 15–19 Aug 2021.

Raaz Dwivedi and Lester Mackey. Generalized kernel thinning. In *International Conference on Learning Representations*, 2022.

Ethan N Epperly and Elvira Moreno. Kernel quadrature with randomly pivoted Cholesky. *arXiv preprint arXiv:2306.03955*, 2023.

David Eriksson and Matthias Poloczek. Scalable constrained Bayesian optimization. In *International Conference on Artificial Intelligence and Statistics*, pp. 730–738. PMLR, 2021.

David Eriksson, Michael Pearce, Jacob Gardner, Ryan D Turner, and Matthias Poloczek. Scalable global optimization via local Bayesian optimization. *Advances in neural information processing systems*, 32, 2019.

Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. Efficient and robust automated machine learning. *Advances in neural information processing systems*, 28, 2015.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep Bayesian active learning with image data. In *International conference on machine learning*, pp. 1183–1192. PMLR, 2017.

Jacob Gardner, Geoff Pleiss, Kilian Q Weinberger, David Bindel, and Andrew G Wilson. GPyTorch: Blackbox matrix-matrix Gaussian process inference with GPU acceleration. In *Advances in Neural Information Processing Systems*, pp. 7576–7586, 2018.

Jacob R Gardner, Matt J Kusner, Zhixiang Eddie Xu, Kilian Q Weinberger, and John P Cunningham. Bayesian optimization with inequality constraints. In *ICML*, volume 2014, pp. 937–945, 2014.

Roman Garnett. *Bayesian optimization*. Cambridge University Press, 2023.

Michael A Gelbart, Jasper Snoek, and Ryan P Adams. Bayesian optimization with unknown constraints. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, pp. 250—-259, 2014. doi: https://doi.org/10.48550/arXiv.1403.5607.

Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

Javier González, Zhenwen Dai, Philipp Hennig, and Neil Lawrence. Batch Bayesian optimization via local penalization. In *Artificial intelligence and statistics*, pp. 648–657. PMLR, 2016.

Siegfried Graf and Harald Luschgy. *Foundations of quantization for probability distributions*. Springer, 2007.

Ryan-Rhys Griffiths, Leo Klarner, Henry Moss, Aditya Ravuri, Sang T Truong, Bojana Rankovic, Yuanqi Du, Arian Rokkum Jamasb, Julius Schwartz, Austin Tripp, et al. GAUCHE: A library for Gaussian processes in chemistry. In *ICML 2022 2nd AI for Science Workshop*, 2022.

Carlos Guestrin, Andreas Krause, and Ajit Paul Singh. Near-optimal sensor placements in Ggaussian processes. In *Proceedings of the 22nd international conference on Machine learning*, pp. 265–272, 2005.

Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2024. URL `https://www.gurobi.com`.

Huong Ha, Vu Nguyen, Hongyu Zhang, and Anton van den Hengel. Provably efficient Bayesian optimization with unbiased Gaussian process hyperparameter estimation. *arXiv preprint arXiv:2306.06844*, 2023.

N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

Satoshi Hayakawa, Harald Oberhauser, and Terry Lyons. Positively weighted kernel quadrature via subsampling. *Advances in Neural Information Processing Systems*, 35:6886–6900, 2022.

Satoshi Hayakawa, Harald Oberhauser, and Terry Lyons. Sampling-based Nyström approximation and kernel quadrature. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 12678–12699, 2023.

Philipp Hennig, Michael A Osborne, and Mark Girolami. Probabilistic numerics and uncertainty in computations. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 471(2179):20150142, 2015.

Philipp Hennig, Michael A Osborne, and Hans P Kersting. *Probabilistic Numerics: Computation as Machine Learning*. Cambridge University Press, 2022.

José Miguel Hernández-Lobato, Matthew W Hoffman, and Zoubin Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. *Advances in neural information processing systems*, 27, 2014.

José Miguel Hernández-Lobato, Michael Gelbart, Matthew Hoffman, Ryan Adams, and Zoubin Ghahramani. Predictive entropy search for Bayesian optimization with unknown constraints. In *International conference on machine learning*, pp. 1699–1707. PMLR, 2015.

José Miguel Hernández-Lobato, Michael A. Gelbart, Ryan P. Adams, Matthew W. Hoffman, and Zoubin Ghahramani. A general framework for constrained Bayesian optimization using information-based search. *Journal of Machine Learning Research*, 17(1):5549–5601, 2016.

José Miguel Hernández-Lobato, James Requeima, Edward O Pyzer-Knapp, and Alán Aspuru-Guzik. Parallel and distributed Thompson sampling for large-scale accelerated exploration of chemical space. In *International conference on machine learning*, pp. 1470–1479. PMLR, 2017.

Matthew D Hoffman and Andrew Gelman. The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. *J. Mach. Learn. Res.*, 15 (1):1593–1623, 2014.

Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011.

Ferenc Huszár and David Duvenaud. Optimally-weighted herding is Bayesian quadrature. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, pp. 377—-386, 2012. doi: https://doi.org/10.48550/arXiv.1204.1664.

Carl Hvarfner, Danny Stoll, Artur Souza, Marius Lindauer, Frank Hutter, and Luigi Nardi. πBO: Augmenting acquisition functions with user beliefs for bayesian optimization. In *International Conference on Learning Representations*, 2022.

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

Ajay J Joshi, Fatih Porikli, and Nikolaos Papanikolopoulos. Multi-class active learning for image classification. In *2009 ieee conference on computer vision and pattern recognition*, pp. 2372–2379. IEEE, 2009.

Motonobu Kanagawa, Bharath K Sriperumbudur, and Kenji Fukumizu. Convergence guarantees for kernel-based quadrature rules in misspecified settings. *Advances in Neural Information Processing Systems*, 29, 2016.

Kirthevasan Kandasamy, Akshay Krishnamurthy, Jeff Schneider, and Barnabás Póczos. Parallelised Bayesian optimisation via Thompson sampling. In *International Conference on Artificial Intelligence and Statistics*, pp. 133–142. PMLR, 2018.

Toni Karvonen. *Kernel-based and Bayesian methods for numerical integration*. PhD thesis, Aalto University, 2019.

Tarun Kathuria, Amit Deshpande, and Pushmeet Kohli. Batched Gaussian process bandit optimization via determinantal point processes. *Advances in Neural Information Processing Systems*, 29, 2016.

Danial Khatamsaz, Brent Vela, Prashant Singh, Duane D Johnson, Douglas Allaire, and Raymundo Arróyave. Bayesian optimization with active learning of design constraints using an entropy-based approach. *npj Computational Materials*, 9(1):49, 2023.

Balhae Kim, Jungwon Choi, Seanie Lee, Yoonho Lee, Jung-Woo Ha, and Juho Lee. On divergence measures for Bayesian pseudocoresets. *Advances in Neural Information Processing Systems*, 35:757–767, 2022.

Andreas Kirsch, Joost Van Amersfoort, and Yarin Gal. BatchBALD: Efficient and diverse batch acquisition for deep Bayesian active learning. *Advances in neural information processing systems*, 32, 2019.

Christine Kiss and Martin Bichler. Identification of influencers—measuring influence in customer networks. *Decision Support Systems*, 46(1):233–253, 2008.

Andreas Krause and Carlos E Guestrin. Near-optimal nonmyopic value of information in graphical models. In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, pp. 324–331, 2012. doi: https://doi.org/10.48550/arXiv.1207.1394.

Jan Kremer, Kim Steenstrup Pedersen, and Christian Igel. Active learning with support vector machines. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4(4):313–326, 2014.

Sanjiv Kumar, Mehryar Mohri, and Ameet Talwalkar. Sampling methods for the Nyström method. *Journal of Machine Learning Research*, 13(Apr):981–1006, 2012.

Norbert Kuschel and Rüdiger Rackwitz. Two basic problems in reliability-based structural optimization. *Mathematical Methods of Operations Research*, 46: 309–333, 1997.

Harold J Kushner. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal of basic engineering*, 86:97 – 106, 1964.

Vidhi Lalchand and Carl Edward Rasmussen. Approximate inference for fully Bayesian Gaussian process regression. In *Symposium on Advances in Approximate Bayesian Inference*, pp. 1–12. PMLR, 2020.

Jean B Lasserre. A new look at nonnegativity on closed sets and polynomial optimization. *SIAM Journal on Optimization*, 21(3):864–885, 2011.

Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne VanBriesen, and Natalie Glance. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 420–429, 2007.

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

Shibo Li, Jeff M Phillips, Xin Yu, Robert Kirby, and Shandian Zhe. Batch multi-fidelity active learning with budget constraints. *Advances in Neural Information Processing Systems*, 35:995–1007, 2022.

Christopher A Lipinski, Franco Lombardo, Beryl W Dominy, and Paul J Feeney. Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. *Advanced drug delivery reviews*, 23(1-3):3–25, 1997.

Dong C Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

ER Logan, Erin M Tonita, KL Gering, Jing Li, Xiaowei Ma, LY Beaulieu, and JR Dahn. A study of the physical properties of Li-ion battery electrolytes containing esters. *Journal of The Electrochemical Society*, 165(2):A21, 2018.

Turab Lookman, Prasanna V Balachandran, Dezhen Xue, and Ruihao Yuan. Active learning in materials science with emphasis on adaptive sampling using uncertainties for targeted design. *npj Computational Materials*, 5(1):21, 2019.

David JC MacKay. Information-based objective functions for active data selection. *Neural computation*, 4(4):590–604, 1992.

Dionysis Manousakas, Zuheng Xu, Cecilia Mascolo, and Trevor Campbell. Bayesian pseudocoresets. *Advances in Neural Information Processing Systems*, 33:14950–14960, 2020.

Angeles Martinez, Federico Piazzon, Alvise Sommariva, and Marco Vianello. Quadrature-based polynomial optimization. *Optimization Letters*, 14:1027–1036, 2020.

Dimitrios Milios, Raffaello Camoriano, Pietro Michiardi, Lorenzo Rosasco, and Maurizio Filippone. Dirichlet-based Gaussian processes for large-scale calibrated classification. *Advances in Neural Information Processing Systems*, 31, 2018.

Masahiro Mochizuki, Shogo D Suzuki, Keisuke Yanagisawa, Masahito Ohue, and Yutaka Akiyama. QEX: target-specific druglikeness filter enhances ligand-based virtual screening. *Molecular Diversity*, 23: 11–18, 2019.

Jonas Mockus. The application of Bayesian methods for seeking the extremum. *Towards global optimization*, 2:117, 1998.

Krikamol Muandet, Kenji Fukumizu, Bharath Sriperumbudur, Bernhard Schölkopf, et al. Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends® in Machine Learning*, 10 (1-2):1–141, 2017.

Elvis Nava, Mojmir Mutny, and Andreas Krause. Diversified sampling for batched Bayesian optimization with determinantal point processes. In *International Conference on Artificial Intelligence and Statistics*, pp. 7031–7054. PMLR, 2022.

Vu Nguyen, Santu Rana, Sunil K Gupta, Cheng Li, and Svetha Venkatesh. Budgeted batch Bayesian optimization. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pp. 1107–1112. IEEE, 2016.

Changyong Oh, Jakub Tomczak, Efstratios Gavves, and Max Welling. Combinatorial Bayesian optimization using the graph Cartesian product. *Advances in Neural Information Processing Systems*, 32, 2019.

Anthony O'Hagan. Bayes–Hermite quadrature. *Journal of statistical planning and inference*, 29(3):245–260, 1991.

Ji Won Park, Samuel Stanton, Saeed Saremi, Andrew Watkins, Henri Dwyer, Vladimir Gligorijevic, Richard Bonneau, Stephen Ra, and Kyunghyun Cho. PropertyDAG: Multi-objective Bayesian optimization of partially ordered, mixed-variable properties for biological sequence design. In *NeurIPS 2022 AI for Science: Progress and Promises*, 2022.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, and Luca Antiga. PyTorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Robert Pinsler, Jonathan Gordon, Eric Nalisnick, and José Miguel Hernández-Lobato. Bayesian batch active learning as sparse subset approximation. *Advances in neural information processing systems*, 32, 2019.

Liva Ralaivola, Sanjay J Swamidass, Hiroto Saigo, and Pierre Baldi. Graph kernels for chemical informatics. *Neural networks*, 18(8):1093–1110, 2005.

Carl Edward Rasmussen, Christopher KI Williams, et al. *Gaussian processes for machine learning*, volume 1. Springer, 2006.

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij B Gupta, Xiaojiang Chen, and Xin Wang. A survey of deep active learning. *ACM computing surveys (CSUR)*, 54(9):1–40, 2021.

Robert R Richardson, Michael A Osborne, and David A Howey. Gaussian process regression for forecasting battery state of health. *Journal of Power Sources*, 357:209–219, 2017.

Christoffer Riis, Francisco Antunes, Frederik Hüttel, Carlos Lima Azevedo, and Francisco Pereira. Bayesian active learning with fully Bayesian Gaussian processes. *Advances in Neural Information Processing Systems*, 35:12141–12153, 2022.

Alessandro Rudi, Ulysse Marteau-Ferey, and Francis Bach. Finding global minima via kernel approximations. *arXiv preprint arXiv:2012.11978*, 2020.

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018.

Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

H Sebastian Seung, Manfred Opper, and Haim Sompolinsky. Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 287–294, 1992.

Il'ya Meerovich Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki*, 7(4):784–802, 1967.

Thomas Spangenberg, Jeremy N Burrows, Paul Kowalczyk, Simon McDonald, Timothy NC Wells, and Paul Willis. The open access malaria box: a drug discovery catalyst for neglected diseases. *PloS one*, 8(6):e62906, 2013.

Bharath K Sriperumbudur, Arthur Gretton, Kenji Fukumizu, Bernhard Schölkopf, and Gert RG Lanckriet. Hilbert space embeddings and metrics on probability measures. *The Journal of Machine Learning Research*, 11:1517–1561, 2010.

Yanan Sui, Alkis Gotovos, Joel Burdick, and Andreas Krause. Safe exploration for optimization with Gaussian processes. In *International conference on machine learning*, pp. 997–1005. PMLR, 2015.

Onur Teymur, Jackson Gorham, Marina Riabiz, and Chris Oates. Optimal quantisation of probability measures using maximum mean discrepancy. In *International Conference on Artificial Intelligence and Statistics*, pp. 1027–1035. PMLR, 2021.

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

Daniel F Veber, Stephen R Johnson, Hung-Yuan Cheng, Brian R Smith, Keith W Ward, and Kenneth D Kopple. Molecular properties that influence the oral bioavailability of drug candidates. *Journal of medicinal chemistry*, 45(12):2615–2623, 2002.

Ed Wagstaff, Saad Hamid, and Michael Osborne. Batch selection for parallelisation of Bayesian quadrature. *arXiv preprint arXiv:1812.01553*, 2018.

Xingchen Wan, Vu Nguyen, Huong Ha, Binxin Ru, Cong Lu, and Michael A. Osborne. Think global and act local: Bayesian optimisation over high-dimensional categorical and mixed search spaces. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pp. 10663–10674, 2021.

Xingchen Wan, Pierre Osselin, Henry Kenlay, Binxin Ru, Michael A. Osborne, and Xiaowen Dong. Bayesian optimisation of functions on graphs. *arXiv preprint arXiv:2306.05304*, 2023.

Christopher Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. *Advances in neural information processing systems*, 13, 2000.

Jian Wu, Saul Toscano-Palmerin, Peter I Frazier, and Andrew Gordon Wilson. Practical multi-fidelity Bayesian optimization for hyperparameter tuning. In *Uncertainty in Artificial Intelligence*, pp. 788–798. PMLR, 2020.

Y-T Wu, Youngwon Shin, Robert Sues, and Mark Cesare. Safety-factor based approach for probability-based design optimization. In *19th AIAA applied aerodynamics conference*, pp. 1522, 2001.

Wenjie Xu, Yuning Jiang, Bratislav Svetozarevic, and Colin Jones. Constrained efficient global optimization of expensive black-box functions. In *International Conference on Machine Learning*, pp. 38485–38498. PMLR, 2023.

Jacky Zhang, Rajiv Khanna, Anastasios Kyrillidis, and Sanmi Koyejo. Bayesian coresets: Revisiting the nonconvex optimization perspective. In *International Conference on Artificial Intelligence and Statistics*, pp. 2782–2790. PMLR, 2021.

Yin-Cong Zhi, Yin Cheng Ng, and Xiaowen Dong. Gaussian processes on graphs via spectral kernel learning. *IEEE Transactions on Signal and Information Processing over Networks*, 2023.

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes in Supplementary]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Not Applicable]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Part I

Appendix

Table of Contents

A Proof of Proposition 1

Proof of Proposition 1. Note that the constraint $|\mathbf{w}|_0 \leq n$ is automatically satisfied when we use the simplex method or its variant. Without this constraint, we have a trivial feasible solution $\mathbf{w} = \mathbf{w}_{\text{cand}}$, so, for the optimal solution $\mathbf{w}_*$, we have $\mathbf{w}_*^\top \left[g(\mathbf{X}_{\text{cand}}) \odot q(\mathbf{X}_{\text{cand}})\right] \geq \mathbf{w}_{\text{cand}}^\top \left[g(\mathbf{X}_{\text{cand}}) \odot q(\mathbf{X}_{\text{cand}})\right]$. Since $\mathbb{E}[\tilde{\mathbf{w}}_{\text{batch}}^\top g(\tilde{\mathbf{X}}_{\text{batch}})] = \mathbf{w}_{\text{batch}}^\top \left[g(\mathbf{X}_{\text{batch}}) \odot q(\mathbf{X}_{\text{batch}})\right] = \mathbf{w}_*^\top \left[g(\mathbf{X}_{\text{cand}}) \odot q(\mathbf{X}_{\text{cand}})\right]$, we obtain the first estimate Eq. (8).

For the latter estimate, we first decompose the error into two parts:

$$\begin{aligned}
&\mathbb{E}\left[\left|\tilde{\mathbf{w}}_{\text{batch}}^\top f(\tilde{\mathbf{X}}_{\text{batch}}) - \mathbf{w}_{\text{cand}}^\top f(\mathbf{X}_{\text{cand}})\right|\right] \\
&\leq \mathbb{E}\left[\left|\tilde{\mathbf{w}}_{\text{batch}}^\top f(\tilde{\mathbf{X}}_{\text{batch}}) - \mathbf{w}_{\text{batch}}^\top f(\mathbf{X}_{\text{batch}})\right|\right] + \left|\mathbf{w}_{\text{batch}}^\top f(\mathbf{X}_{\text{batch}}) - \mathbf{w}_{\text{cand}}^\top f(\mathbf{X}_{\text{cand}})\right|.
\end{aligned} \tag{10}$$

For the first term, considering each $x \in \mathbf{X}_{\text{batch}}$ on whether or not it gets included in $\tilde{\mathbf{X}}_{\text{batch}}$, we have

$$\begin{aligned}
&\mathbb{E}\left[\left|\tilde{\mathbf{w}}_{\text{batch}}^\top f(\tilde{\mathbf{X}}_{\text{batch}}) - \mathbf{w}_{\text{batch}}^\top f(\mathbf{X}_{\text{batch}})\right|\right] \\
&\leq \mathbf{w}_{\text{batch}}^\top \left[|f|(\mathbf{X}_{\text{batch}}) \odot (1-q)(\mathbf{X}_{\text{batch}})\right] \leq \mathbf{w}_{\text{batch}}^\top (1-q)(\mathbf{X}_{\text{batch}}) \max_{x \in \mathbf{X}_{\text{batch}}} |f(x)| \\
&= \left[1 - \mathbf{w}_{\text{batch}}^\top q(\mathbf{X}_{\text{batch}})\right] \max_{x \in \mathbf{X}_{\text{batch}}} |f(x)| \leq \left[1 - \mathbf{w}_{\text{cand}}^\top q(\mathbf{X}_{\text{cand}})\right] \max_{x \in \mathbf{X}_{\text{batch}}} |f(x)|,
\end{aligned}$$

where the last inequality follows from the inequality constraint $(\mathbf{w} - \mathbf{w}_{\text{cand}})^\top q(\mathbf{X}_{\text{cand}}) \geq 0$ in the LP. Since $|f(x)| = |\langle f, K_{\text{LP}}(\cdot, x)\rangle| \leq \|f\| K_{\text{LP}}(x, x)^{1/2}$ from the reproducing property of RKHS, we obtain

$$\mathbb{E}\left[\left|\tilde{\mathbf{w}}_{\text{batch}}^\top f(\tilde{\mathbf{X}}_{\text{batch}}) - \mathbf{w}_{\text{batch}}^\top f(\mathbf{X}_{\text{batch}})\right|\right] \leq \epsilon_{\text{rej}} K_{\max} \|f\|. \tag{11}$$

Let us then bound the second term of the RHS of Eq. (10). Note that, from the formula of worst-case error of kernel quadrature (see, e.g., (Hayakawa et al., 2022, Eq. (14))), we can bound

$$\left|\mathbf{w}_{\text{batch}}^\top f(\mathbf{X}_{\text{batch}}) - \mathbf{w}_{\text{cand}}^\top f(\mathbf{X}_{\text{cand}})\right|^2 \leq \|f\|^2 (\mathbf{w}_* - \mathbf{w}_{\text{cand}})^\top K_{\text{LP}}(\mathbf{X}_{\text{cand}}, \mathbf{X}_{\text{cand}})(\mathbf{w}_* - \mathbf{w}_{\text{cand}}) \tag{12}$$

(recall $\mathbf{w}_*$ has the same dimension as $\mathbf{w}_{\text{cand}}$). We now want to estimate

$$(\mathbf{w}_* - \mathbf{w}_{\text{cand}})^\top K_{\text{LP}}(\mathbf{X}_{\text{cand}}, \mathbf{X}_{\text{cand}})(\mathbf{w}_* - \mathbf{w}_{\text{cand}}).$$

Consider approximating K_{LP} by K_{nys}. Since $K_{\text{LP}} - K_{\text{nys}}$ is positive semi-definite from the property of Nyström approximation (see, e.g., the proof of (Hayakawa et al., 2022, Corollary 4)), for any $x, y \in \mathbf{X}_{\text{cand}}$, we have

$$|(K_{\text{LP}} - K_{\text{nys}})(x, y)| \leq |(K_{\text{LP}} - K_{\text{nys}})(x, x)|^{1/2} |(K_{\text{LP}} - K_{\text{nys}})(y, y)|^{1/2} \leq \epsilon_{\text{nys}}^2.$$

Thus, we have

$$\begin{aligned}
&(\mathbf{w}_* - \mathbf{w}_{\text{cand}})^\top \left[(K_{\text{LP}} - K_{\text{nys}})(\mathbf{X}_{\text{cand}}, \mathbf{X}_{\text{cand}})\right](\mathbf{w}_* - \mathbf{w}_{\text{cand}}) \\
&\leq (\mathbf{w}_* + \mathbf{w}_{\text{cand}})^\top (\epsilon_{\text{nys}}^2 \mathbf{1}\mathbf{1}^\top)(\mathbf{w}_* + \mathbf{w}_{\text{cand}}) = 4\epsilon_{\text{nys}}^2.
\end{aligned} \tag{13}$$

Finally, we estimate

$$\begin{aligned}
&(\mathbf{w}_* - \mathbf{w}_{\text{cand}})^\top K_{\text{nys}}(\mathbf{X}_{\text{cand}}, \mathbf{X}_{\text{cand}})(\mathbf{w}_* - \mathbf{w}_{\text{cand}}) \\
&= (\mathbf{w}_* - \mathbf{w}_{\text{cand}})^\top \sum_{j=1}^{n-2} \mathbf{1}_{\{\lambda_j > 0\}} \lambda_j^{-1} \varphi_j(\mathbf{X}_{\text{cand}}) \varphi_j(\mathbf{X}_{\text{cand}})^\top (\mathbf{w}_* - \mathbf{w}_{\text{cand}}) \\
&= \sum_{j=1}^{n-2} \mathbf{1}_{\{\lambda_j > 0\}} \lambda_j^{-1} \left[(\mathbf{w}_* - \mathbf{w}_{\text{cand}})^\top \varphi_j(\mathbf{X}_{\text{cand}})\right]^2.
\end{aligned} \tag{14}$$

From the inequality constraint in the LP, we have $|(\mathbf{w}_* - \mathbf{w}_{\text{cand}})^\top \varphi_j(\mathbf{X}_{\text{cand}})| \leq \epsilon_{\text{LP}} \sqrt{\lambda_j / (n-2)}$, so that Eq. (14) is further bounded as

$$(\mathbf{w}_* - \mathbf{w}_{\text{cand}})^\top K_{\text{nys}}(\mathbf{X}_{\text{cand}}, \mathbf{X}_{\text{cand}})(\mathbf{w}_* - \mathbf{w}_{\text{cand}}) \leq \sum_{j=1}^{n-2} \mathbf{1}_{\{\lambda_j > 0\}} \lambda_j^{-1} \epsilon_{\text{LP}}^2 \frac{\lambda_j}{n-2} \leq \epsilon_{\text{LP}}^2. \tag{15}$$

By adding the both sides of Eqs. (13) and (15), we obtain

$$(\mathbf{w}_* - \mathbf{w}_{\text{cand}})^\top K_{\text{LP}}(\mathbf{X}_{\text{cand}}, \mathbf{X}_{\text{cand}})(\mathbf{w}_* - \mathbf{w}_{\text{cand}}) \leq 4\epsilon_{\text{nys}}^2 + \epsilon_{\text{LP}}^2 \leq (2\epsilon_{\text{nys}} + \epsilon_{\text{LP}})^2.$$

By applying this to Eq. (12), we have $\left|\mathbf{w}_{\text{batch}}^\top f(\mathbf{X}_{\text{batch}}) - \mathbf{w}_{\text{cand}}^\top f(\mathbf{X}_{\text{cand}})\right| \leq \|f\|(2\epsilon_{\text{nys}} + \epsilon_{\text{LP}})$. Combining this with Eqs. (10) and (11) yields the desired inequality Eq. (9). □

B Background

B.1 Gaussian process

GP (Rasmussen et al., 2006) is a widely used Bayesian regression model and the most popular surrogate model in PN. We consider the probabilistic model $\mathbb{P}(\boldsymbol{y}|\boldsymbol{x}, \theta)$ parameterized by $\theta \in \Theta$, mapping from inputs $x \in \mathcal{X}$ to a distribution over outputs/labels $y \in \mathcal{Y}$. Here, the labels can potentially only be observed through a noisy estimate, $y = f(x) + \epsilon$, where y is a continuous value and the noise $\epsilon \sim \mathcal{N}(0, \lambda^2)$ is assumed to be generated by i.i.d. zero-mean Gaussian, and λ^2 is the noise variance. Given a labelled dataset $\mathcal{D}_0 = \{\boldsymbol{x}_n, \boldsymbol{y}_n\}_{n=1}^N := (\mathcal{X}_N, \mathcal{Y}_N)$, GP regression model is given by $f \mid D_0 \sim \mathcal{GP}(m, C)$, where

$$\begin{aligned} m(x) &= K(x, \boldsymbol{x}_0)\boldsymbol{K}_\lambda^{-1}\boldsymbol{y}_0, \\ C(x, x') &= K(x, x') - K(x, \boldsymbol{x}_0)\boldsymbol{K}_\lambda^{-1}K(\boldsymbol{x}_0, x'), \end{aligned} \tag{16}$$

f is the surrogate function, $m(\cdot)$ and $C(\cdot, \cdot)$ are the mean and covariance of posterior predictive distribution of f, and $\mathcal{D}_0 = (\boldsymbol{x}_0, \boldsymbol{y}_0)$ is the observed dataset. K is the kernel parameterized by θ[13] and $\boldsymbol{K}_\lambda^{-1} := [K(\boldsymbol{x}_0, \boldsymbol{x}_0) + \lambda^2\boldsymbol{I}]^{-1}$, and $\lambda^2\boldsymbol{I}$ is the diagonal likelihood variance matrix.

B.2 Fully Bayesian Gaussian Process

In this paper, we will consider the Bayesian active learning for Fully Bayesian Gaussian Process (FBGP) (Riis et al., 2022). FBGP extends a GP by placing a prior over the hyperparameter $\mathbb{P}(\theta)$ and approximating their full posteriors. The predictive posterior for the test inputs x^* is

$$\begin{aligned} &\mathbb{P}(y^* \mid x^*, \mathcal{D}_0) \\ &= \iint \mathbb{P}(y^* \mid f^*, \theta, x^*, \mathcal{D}_0)\mathrm{d}\mathbb{P}(f^* \mid \theta, x^*, \mathcal{D}_0)\mathrm{d}\mathbb{P}(\theta \mid \mathcal{D}_0). \end{aligned}$$

While the inner integral for f reduces to the normal GP predictive posterior, the outer integral for θ remains intractable and typically approximated by MCMC.

B.3 Bayesian Quadrature

Bayesian quadrature (BQ) is an algorithm for evaluating integrals given by:

$$\hat{Z} = \int_{\mathcal{X}} f(x)\mathrm{d}\pi(x), \tag{17}$$

where f is the black-box function we wish to integrate against a known probability measure π. The difference from BO is the objective being integration, not global optimisation. The integration problem is widely recognised in statistical learning: expectations, variances, marginalisation, ensembles, Bayesian model selection, and Bayesian model averaging. BQ is, like BO, solved by GP-surrogate-model-based active learning. The batch acquisition methods are also shared with batch BO. The methodological differences are:

1. BQ typically assumes a specific kernel to make the integration analytical (e.g. RBF kernel).
2. While BO requires to approximate the black-box function only in the vicinity of the global optimum, BQ needs to approximate the whole region of interest defined by the probability measure π.

Thus, BQ is a purely explorative algorithm, and the uncertainty sampling acquisition function is often applied.

The classic method to estimate the integral exploits Gaussianity. Let π be multivariate normal distribution $\pi(x) = \mathcal{N}(x; \mu_\pi, \boldsymbol{\Sigma}_\pi)$, and the kernel K be RBF kernel, which can be represented as Gaussian $K(\boldsymbol{x}_0, x) = v\sqrt{|2\pi\mathbf{W}|}\mathcal{N}(\boldsymbol{x}_0; x, \mathbf{W})$, where v is kernel variance and $\mathbf{W}$ is the diagonal covariance matrix whose diagonal elements are the lengthscales of each dimension. As the product of two Gaussians is a Gaussian, the integrand becomes a Gaussian and its integral has the closed form, as such:

$$\begin{aligned} \int m(x)\pi(x)dx &= v\left[\int \mathcal{N}(x; \boldsymbol{x}_0, \mathbf{W})\mathcal{N}(x; \mu_\pi, \boldsymbol{\Sigma}_\pi)dx\right]^\top \boldsymbol{K}_\lambda^{-1}\boldsymbol{y}_0, && (18) \\ &= v\left[\int \mathcal{N}(x; \boldsymbol{x}_0, \mathbf{W})\mathcal{N}(x; \mu_\pi, \boldsymbol{\Sigma}_\pi)dx\right]^\top \boldsymbol{K}_\lambda^{-1}\boldsymbol{y}_0, && (19) \\ &= v\mathcal{N}(\boldsymbol{x}_0; \mu_\pi, \mathbf{W} + \boldsymbol{\Sigma}_\pi)^\top \boldsymbol{K}_\lambda^{-1}\boldsymbol{y}_0, && (20) \\ &= \boldsymbol{z}^\top \boldsymbol{K}_\lambda^{-1}\boldsymbol{y}_0 && (21) \end{aligned}$$

[13] We typically assume zero mean GP prior over function space $\mathcal{GP}(0, K)$, and we assume Gaussian likelihood $\mathcal{N}(0, \lambda^2)$, hence the resulting posterior distribution is closed-form as shown in Eq. (16), thanks to Gaussianity. Throughout the paper, we refer to a symmetric positive semi-definite kernel just as a kernel.

where $\boldsymbol{z} := v\mathcal{N}(\boldsymbol{x}_0; \mu_\pi, \mathbf{W} + \boldsymbol{\Sigma}_\pi)$. As such, the integration of GP over the measure π is analytical. This $\boldsymbol{z}$ corresponds to the kernel mean in Eqs. (3)-(4). This clearly explains that we need an analytical kernel mean to perform BQ. Thus, classical BQ methods have limitations on prior and kernel selections to be analytical. To make the integration closed-form, the prior needs to be uniform or Gaussian, and the kernel also needs to be limited selection (e.g. RBF kernel, see Table 1 in Briol et al. (2019)). Recent work (Adachi et al., 2022) extends this to arbitrary kernel and prior.

C Related Work

Batch Active Learning A wide variety of batch methods has been proposed for each task: batch AL (Pinsler et al., 2019; Kirsch et al., 2019; Riis et al., 2022), batch BQ (Wagstaff et al., 2018; Adachi et al., 2022, 2023b) and batch BO, a greedy extension of sequential algorithms (Azimi et al., 2010; González et al., 2016; Eriksson et al., 2019; Balandat et al., 2020), diversified batch with determinantal point process (DPP) (Kathuria et al., 2016; Nava et al., 2022). Constrained batch construction has been researched in BO community (Hernández-Lobato et al., 2016; Letham et al., 2019; Eriksson & Poloczek, 2021). However, most works do not discuss the relationship to quality of batch construction like KQ methods.

Kernel Quadrature For general KQ methods, There are a number of KQ algorithms; herding/optimization (Chen et al., 2010; Bach et al., 2012; Huszár & Duvenaud, 2012), random sampling (Bach, 2017; Belhadji et al., 2019), DPP (Belhadji et al., 2019; Belhadji, 2021), kernel thinning (Dwivedi & Mackey, 2021, 2022), recombination (Hayakawa et al., 2022, 2023), kernel Stein discrepancy (Chen et al., 2018, 2019; Teymur et al., 2021), randomly pivoted Cholesky (Epperly & Moreno, 2023). Similarly, Bayesian coresets is one of applications of quantization method and proposes a variety of algorithms (Campbell & Broderick, 2019; Manousakas et al., 2020; Zhang et al., 2021; Chen et al., 2022), thus they are strongly related to KQ. KQ is a more proper framework for GP-based AL as it can incorporate the model uncertainty information for batch construction.

Adaptive Batch Size While all of the above KQ/Bayesian coresets methods can be used for batch construction, almost all methods assume the batch size is predefined. The adaptive batch size setting remains largely unsolved. In batch BO, Nguyen et al. (2016) firstly formulated this setting as a Gaussian mixture fitting to the acquisition function and estimated the batch size as Bayesian model selection, which is obviously non-KQ-based. No other work proposes dynamic batch-size AL, to the best of our knowledge.

Connection to Bayesian Coresets While Bayesian coresets use Kullback-Leibler or Wasserstein divergence as a metric (Kim et al., 2022) and weighted Euclidean inner product, KQ uses MMD. MMD and KQ have a direct relationship and KQ can incorporate additional information on model uncertainty to quantize the probability distribution. In KQ, the selected batch points are chosen to reduce the model uncertainty, which is advantageous property for active learning that needs to train model effectively. Thus, KQ can utilize more information if the model has the analytical predictive covariance like GP. The quality of batch construction can be evaluated as the worst-case error in Eq. (2), as this is equivalent to the MMD, which evaluates the divergence between quantized probability measure and the target distribution.

D Batch Bayesian Active Learning as Quantization

D.1 Sparse Subset Approximation

Pinsler et al. (2019) proposed the batch construction heuristics with sparse subset approximation. The original paper did not state this but the formulaton is exactly the same as the weighted quantization task. They interpret this as Bayesian coresets (Campbell & Broderick, 2019), which is another view of weighted quantization and essentially close to KQ. Their attempt is simple: they construct batch samples to best approximate the true posterior $\mathbb{P}(\theta | \mathcal{D}_0 \cup \mathcal{D}_t) \approx \mathbb{P}(\theta | \mathcal{D}_0 \cup \mathcal{D}_N)$. Here, the true posterior $\mathbb{P}(\theta | \mathcal{D}_0 \cup \mathcal{D}_N)$ means the posterior with the complete dataset $\mathcal{D}_N = (\mathcal{X}_N, \mathcal{Y}_N)$. Unsurprisingly, this true posterior is not available in the AL setting. While an unlabelled pool of candidates $\mathcal{X}_N$ is given, $\mathcal{Y}_N$ is not given. $\mathcal{Y}_N$ means the exhaustive number of costly human labelling, which we wish to reduce and is the motivation to perform AL. Hence, we have to approximate the true posterior. Pinsler et al. (2019) approximated the true posterior using the expectation of the current posterior with respect to the current predictive distribution.

$$\mathbb{P}(\theta \mid \mathcal{X}_N, \mathcal{D}_0) = \int \mathbb{P}(\theta \mid \mathcal{D}_0, \mathcal{X}_N, y^*) \mathrm{d}\mathbb{P}(y^* \mid \mathcal{D}_0, \mathcal{X}_N),$$

where $\mathbb{P}(y^* \mid \mathcal{D}_0, \mathcal{X}_N)$ is the predictive posterior for the unlabelled inputs $\mathcal{X}_N$. This *expected* posterior is different from the current posterior $\mathbb{P}(\theta \mid \mathcal{D}_0)$ as it is explicitly conditioned on $\mathcal{X}_N$. In other words, the current posterior is not conditioned on the input x, thus it is not useful to guide the next query $\mathcal{X}_t$. This expected posterior is conditioned on the input, hence this can guide the next query points. They used this expected posterior as the target distribution in the quantization task, then constructed the batch samples using Bayesian coresets. Namely, approximating expected posterior using the subset $\mathcal{X}_t \subset \mathcal{X}_N$, $\mathbb{P}(\theta \mid \mathcal{X}_N, \mathcal{D}_0) \approx \mathbb{P}(\theta \mid \mathcal{X}_t, \mathcal{D}_0)$. This heuristic outperformed popular baselines, such as BALD (Houlsby et al., 2011) and clustering-based approach.

D.2 Reinterpret as Kernel Quadrature

We reinterpret this formulation as a KQ task.

1. The target distribution μ is the weighted samples $(\mathbf{w}_{\text{cand}}, \mathbf{X}_{\text{cand}})$, where $\mathbf{X}_{\text{cand}} := \mathcal{X}_N$ is the candidate samples (the unlabelled pool), and $\mathbf{w}_{\text{cand}} \propto \mathbb{P}(\theta \mid x_l, \mathcal{D}_0)$ is the weights of candidate samples and $x_l \in \mathbf{X}_{\text{cand}}$. The weights are the expected posterior as introduced in the section D.1.
2. The quantized distribution ν is the weighted samples $(\mathbf{w}_{\text{batch}}, \mathbf{X}_{\text{batch}})$, and $\mathbf{X}_{\text{batch}} \subset \mathbf{X}_{\text{cand}}$ is the next batch query points.
3. The kernel is the *expected* predictive covariance of FBGP model, $K(\cdot, \cdot) := \mathbb{E}_{\theta \sim \mathbb{P}(\theta \mid \mathcal{D}_0)}[C(\cdot, \cdot \mid \theta)]$. This enables us to incorporate the model uncertainty to construct the quantized samples unlike the original paper (Pinsler et al., 2019). The original work used the weighted Euclidean inner product using only the expected posterior, which loses the information of uncertainty.

As such, we can reinterpret the sparse subset approximation method as a KQ task. Hence, we can apply our LP formulation in the AdaBatAL to batch AL tasks. Furthermore, we have additional room for incorporating the information in LP formulation; the reward g. We used the B-QBC acquisition function as the reward for incorporating the additional information on estimation variance in mean estimation.

E Batch Bayesian Optimization as Quantization

E.1 Primer of Bayesian Optimization

BO (Mockus, 1998; Garnett, 2023) aims to optimize the blackbox function f when there is no access to the closed-form function nor gradient but can query the function pointwise.

$$x^*_{\text{true}} = \operatorname*{argmax}_{x \in \mathcal{X}} f(x), \tag{22}$$

where x^*_{true} is the ground truth of the global optimum. We wish to find as large $f(x)$ as possible under some given budget, such as the overall cost or number of queries. BO is a surrogate-model-based optimizer, which typically adopts GP. BO is also extended to batch BO. The core difference between BO and BQ is that BO only needs an accurate model in the vicinity of global optimal locations, whereas BQ needs an accurate model all over the domain. Therefore, BQ can be viewed as a pure exploration algorithm (as BQ explores based on only uncertainty, as shown in Eq. (5)).

E.2 Batch Bayesian Optimization as a Kernel Quadrature

Adachi et al. (2023a) has introduced the heuristics to recast the batch BO as a KQ task:

$$\delta_{x^*_{\text{true}}} \in \operatorname*{argmax}_{\pi} \int f(x) \mathrm{d}\pi(x), \tag{23}$$

where δ_x is the delta distribution at x and $\pi(x) := \mathbb{P}(f(x) = \max_{x \in \mathcal{X}} f(x))$ is a probability distributions (belief) of x^*_{true}. Note that the optimization target in Eq. (23) has now changed from x into π. We view Eq. (23) as a batch-sequential 'measure' (π) optimization, which updates π over each iteration. The more data we observe, the more confidently we can estimate the location of the global maximum. This corresponds to that the distribution π 'shrinks' toward the true global optimum location, and becomes the delta distribution in the ideal case of a single global maximum. The role of π can be understood as 'exploitation', which determines the promising region over the domain.

The intuition of this reformulation is as follows:

(a) Measure optimization is *dual* to original global optimization (Lasserre, 2011). Classically, a deterministic polynomial regressor (Lasserre, 2011; Martinez et al., 2020; Rudi et al., 2020) has been applied with provable convergence rate.
(b) Measure optimization is *convex optimization* with a linear objective function even if the function f is non-convex (Rudi et al., 2020).
(c) Convexity negates the necessity to maximize the non-convex multimodal acquisition function. Thus, it provides a computationally efficient solution without worrying acquisition function being properly maximised at each iteration and batch in typical batch heuristics.
(d) Variance of π correlates with the variance of predictive distribution under π (Adachi et al., 2023a). Thus, minimising the variance of π leads to minimising the GP predictive variance under π, which is exactly the BQ task and finding the batch points is exactly the KQ task via the duality in Eq. (5)
(e) Unlike typical BO, BQ has the robustness guarantee when RKHS is misspecified (Kanagawa et al., 2016; Hayakawa et al., 2022). This is advantageous as GP in BO tends to be suboptimally tuned (Ha et al., 2023).

This heuristic approach showed the state-of-the-art performance as batch BO over commonly used 8 heuristics of batch BO(Adachi et al., 2023a). Similar to the fact that many acquisition functions have been proposed, a variety of π definitions can be adopted. Adachi et al. (2023a) has proposed two variants: (i) Thompson sampling (TS) (Thompson, 1933), $\pi := \mathbb{P}(x^* | \boldsymbol{D}_t)$, where $x^* := \text{argmax}_x f$ is the maximum location of f, a sample from GP predictive posterior distribution, $\boldsymbol{D}_t := (\boldsymbol{x}_{\text{obs, t}}, \boldsymbol{y}_{\text{obs, t}})$ is the dataset we observed until t-th iterations. That is, we use the current belief of the maximum location of surrogate model x^* instead of ground truth x^*_{true}. (ii) Probability of improvement (PI) (Kushner, 1964): $\pi := \mathbb{P}(f \geq \eta | \boldsymbol{D}_t)$, where $\eta := \text{argmax}_x \mathbf{D}_t$. In both cases, π shrinks toward x^*_{true} upon $\boldsymbol{D}_t$ updates.

E.3 Defining as Kernel Quadrature

We define this formulation as a KQ task.

1. The target distribution μ is the weighted samples $(\mathbf{w}_{\text{cand}}, \mathbf{X}_{\text{cand}})$, where $\mathbf{X}_{\text{cand}} \sim \pi(x)$ is the candidate samples drawn from the probability distribution of x_{true}. $\mathbf{w}_{\text{cand}}$ can be equal weights or importance weights when proposal distribution is applied (Adachi et al., 2023a).
2. The quantized distribution ν is the weighted samples $(\mathbf{w}_{\text{batch}}, \mathbf{X}_{\text{batch}})$, and $\mathbf{X}_{\text{batch}} \subset \mathbf{X}_{\text{cand}}$ is the next batch query points.
3. The kernel is the predictive covariance of normal GP, $K(\cdot, \cdot) := C(\cdot, \cdot \mid \theta)$.

As such, regardless of the π definition, batch BO can be solved as a KQ task.

E.4 Constrained Bayesian Active Learning and Optimization

Suppose we have constraints with which we must comply, but we do not know the constraint functions a priori and are only observable pointwise. We can model such constraint functions by GPs, similarly to the surrogate model of the objective function. We can classify the type of constraints into (A) continuous, and (B) binary constraints.

E.4.1 Modelling Continuous Constraints

Continuous constraints naturally appear in the form of threshold (e.g. controlling a car not to exceed the speed limit, or maximize the computer power not to exceed the temperature limit), given by:

$$Q_\ell(x) \geq 0 \tag{24}$$

where Q_ℓ is the ℓ-th latent constraint function. We can reformulate most constraints to be the form of Eq. (24). For instance, when we wish for the temperature not to surpass the limit $T_{\text{limit}} \geq T$, we can set $Q_\ell(T) = T_{\text{limit}} - T$.

We assume we can query these latent values $g_\ell(x)$ at the designated location x, but the function itself is unknown. We need to guess the function shape only from queries.

We place a GP regression model on the latent values $Q_\ell(x)$. Then, the probability of constraint satisfaction q_ℓ can be given:

$$q_\ell(x) := \mathbb{P}(Q_\ell(x) \geq 0) = \Phi\left(\frac{m_\ell(x)}{\sqrt{C_\ell(x, x)}}\right) \tag{25}$$

where m_ℓ and C_ℓ are the posterior predictive mean and covariance of GP on the ℓ-th constraint, $\Phi(x)$ is the cumulative distribution function of the standard normal distribution $\mathcal{N}(x; 0, 1)$.

E.4.2 Modelling Binary Constraints

Binary constraints return the constraint satisfaction as a Boolean value (yes or no). This is typically modelled with a GP classifier: As the classification likelihood such as Bernoulli likelihood is not conjugate with GP prior, the resulting posterior predictive distribution is no more closed-form. As such, we normally estimate the posterior predictive distribution via sampling functions from latent space, then transform them via the so-called link function Rasmussen et al. (2006).

We adopted an approach with Dirichlet-based GP (DGP) (Milios et al., 2018) for scalability. Let $f_\ell \sim \mathcal{GP}(m_\ell, C_\ell)$ be the GP classifier modelling the ℓ-th binary constraint, the binary feedback $y = 1$ be the constraint satisfaction, $y = 0$ be the constraint violation. Monte Carlo integration via transforming the sampled function with the link function can estimate the expectation of binary probability as Bernoulli distribution:

$$q_\ell(x) := \mathbb{P}(y = 1 \mid x) = \int \frac{\exp(f_{\ell,i})}{\sum \exp(f_{\ell,i})} \mathbb{P}(f_{\ell,i} | x, \mathbf{D}_\ell) \mathrm{d} f_\ell \tag{26}$$

where $\mathbf{D}_\ell$ is the observed dataset of the constraint satisfaction $\mathbf{y}_\ell$ at the inputs $\mathbf{x}_\ell$.

E.4.3 Constrainted Active learning and Optimization

With the above constraint models, the typical constrained BO is performed by constraining the acquisition function, namely, $\alpha(x) \prod_{\ell=1}^{c} q_\ell(x)$ (Gardner et al., 2014; Gelbart et al., 2014). In an AdaBatAl algorithm, we can incorporate the constraint information as q in the LP formulation. We do not need to modify the acquisition function.

F Kernel Quadrature for Intractable Kernel Mean

Table 1: How to set the target distribution for each active learning task.

task	target distribution	meaning
Active learning	$\mathcal{X}_N$	unlabelled pool of candidate inputs
Bayesian optimization	$\mathbb{P}(f(x) = \max_{x \in \mathcal{X}} f(x))$	probability distribution of global optimum location
Bayesian quadrature	$\mathbb{P}(x)$	prior distribution

Table 1 summarizes the target distribution definitions for each AL, BO, and BQ task. While active learning considers the discrete candidates, BO and BQ consider continuous distributions. As seen in the section B.3, only a handful of combinations of continuous target distributions and the kernels can provide the analytical kernel mean and variance, thereby providing the analytical expectation of test functions in the section 3.3. This is not always true, particularly for intractable probability distribution (e.g. Thompson sampling in BO), and/or intractable kernel (e.g. Tanimoto kernel for drug discovery tasks). We review how to construct a KQ task for such an intractable pair of target distribution and kernel in this section.

Intractable Expectations of Test Functions We consider *approximately* constructing the probability measure to estimate the expectation. We construct *empirical measure* $\mu_{\text{cand}}(x) := \sum_{i=1}^{P} w_i \delta_{x_i} := (\mathbf{w}_{\text{cand}}, \mathbf{X}_{\text{cand}})$, where $\mathbf{X}_{\text{cand}} \subset \mathbf{X}^N$. We draw very large N samples to approximate the expectations, which assumes N is sufficiently larger than the batch size n, $N \gg n$. Hence, we can approximate $\int_{\mathcal{X}} \boldsymbol{\varphi}(x)\mathrm{d}\mu(x) \approx \mathbf{w}_{\text{cand}}^T \boldsymbol{\varphi}(\mathbf{X}_{\text{cand}})$. This permits kernel quadrature for any pair (K, μ), unlike the original BQ. If directly sampling from $\mu(x)$ is expensive, the empirical measure can be constructed with importance weights. Namely, we draw large samples from cheaper-to-sample distribution (e.g., domain $\mathcal{X}$), then calculate the importance weights by taking the ratio of the probability density function (see details in (Adachi et al., 2023a)).

Error Bounds While the empirical measure makes BQ/KQ applicable to an arbitrary combination of (K, μ), this produces an additional approximation error. The total error bounds of this KQ method are given by:

$$\text{wce}(Q_{\boldsymbol{w}}) \leq 2 \sup_{x,y} \sqrt{K(x,y) - K_0(x,y)} + \text{MMD}_{\mathcal{H}}(\mu, \mu_{\text{cand}}). \tag{27}$$

The proof is given in Proposition 1 in (Hayakawa et al., 2022). The first and second terms correspond to the Nyström and the empirical measure approximation error, respectively. When we take large M Nyström samples and N empirical measures, we can make error bounds tighter if the time budget allows. In practice, we take $N = 20,000$ and $M = 500$ for the batch size $n \leq 100$. (see Appendix in (Adachi et al., 2022) for how to set these values).

G Experimental Details

G.1 Training details

We have tested AdaBatAL for 7 synthetics and 7 real-world tasks for batch AL and BO tasks. Our experiments were repeated 10 times and took a mean and one standard error with different random seeds (the seeds are shared with baseline methods). We use FBGP for batch AL tasks, and simple GP with type-II maximum likelihood estimation for batch BO tasks. The kernel is different for each task but shared with baseline methods (see details in the dataset section). We randomly generated 10 samples as the initial dataset $\mathcal{D}_0$. We use different batch sizes for each task (see details in the dataset section). While the fixed batch size methods simply adopt this as the batch size, AdaBatAL sets this as the upper bound of batch sizes. This means the AdaBatAL tends to query a smaller number of samples than fixed batch size methods. We iterated this batch acquisition process for the fixed iteration times and compared the best-observed values at the last round. For the fair comparison with the adaptive batch size method, we employ the accumulated queries as the metric, which counts the total number of queries at the t-th iteration. As explained, AdaBatAL yields the smaller accumulated queries with the same iteration times. For constrained cases, we removed the violated samples. Thus, constrained tasks yield smaller accumulated queries than unconstrained cases even with the same batch sizes and the same iteration times. Surprisingly, non-adaptive batch baselines tend to have smaller batch sizes than adaptive AdaBatAL due to constraint violation (See Figure 5).

Our code is built upon PyTorch-based libraries (Paszke et al., 2019; Gardner et al., 2018; Balandat et al., 2020; Griffiths et al., 2022) and Gurobipy (Gurobi Optimization, LLC, 2024) is used to solve the linear programming. All baseline methods are official implementations in BoTorch or coded with BoTorch (Balandat et al., 2020).

Batch Bayesian Optimization We use a constant-mean GP with either RBF, Tanimto, or graph diffusion kernel for batch BO tasks. In each iteration of the active learning loop, the outputs are standardized to have zero mean and unit variance. We optimize the hyperparameter by maximizing the marginal likelihood (type-II maximum likelihood estimation) using L-BFGS-B optimizer (Liu & Nocedal, 1989) implemented with BoTorch (Balandat et al., 2020). The initial data sets consist of ten data points drawn by Sobol sequence (Sobol', 1967), and in each iteration, multiple data points are queried as the batch acquisition (upper bound for AdaBatAL). We adopt log regret if the true global maxima are known, otherwise, the log of best-observed value is the evaluation metric using the test dataset. The models are implemented in GPyTorch (Gardner et al., 2018). All experiments are repeated ten times with different initial data sets via different random seeds.

Batch active learning We use a zero-mean GP with an RBF kernel for all batch AL tasks. In each iteration of the active learning loop, the inputs are rescaled to the unit cube $[0, 1]^d$, and the outputs are standardized to have zero mean and unit variance. Following Lalchand & Rasmussen (2020), we give all the hyperparameters relatively uninformative $\mathcal{N}(0, 3)$ lognormal priors. The initial data sets consist of ten data points drawn by Sobol sequence (Sobol', 1967), and in each iteration, 10 data points are queried as the batch acquisition (upper bound for AdaBatAL). The unlabeled pool consists of the 10,000 data points drawn by Sobol sequence all over the domain. We used this unlabelled pool and corresponding true values as the test dataset for the evaluation. We adopt negative log marginal likelihood (NLML) as the evaluation metric using the test dataset. The inference in FBGP is carried out using NUTS (Hoffman & Gelman, 2014) in Pyro (Bingham et al., 2019) with five chains and 500 samples, including a warm-up period with 200 samples. The remaining 1500 samples are all used for the acquisition functions. The models are implemented in GPyTorch (Gardner et al., 2018). All experiments are repeated ten times with different initial data sets via different random seeds.

For batch AL, we typically assume training a model is very expensive (e.g. deep learning). FBGP is expensive to train even with parallel chains. Thus, we exclude methods like hallucination that require the sequential update of the model to select multiple points. This assumption is widely shared with the AL community (e.g. Kirsch et al. (2019); Pinsler et al. (2019)). Moreover, all baseline batch AL methods do not consider probabilistic constraints. We simply follow the constrained BO approaches, explained in section E.4.3.

Extension to non-continuous input domain Almost all methods are not compatible with categorical and mixed input spaces due to the continuity assumption in these methods. To enable comparison against these methods, we adopt the nearest neighbor in discrete or mixed problems: namely, we optimise the discrete variables as bounded continuous variables, then the selected continuous locations are classified into the closest original discrete values. For the graph space, we deem the search space itself to be a graph and the objective is to find a subgraph. This is different from, for example, the drug discovery problem, whose input variables are graphs but the space itself is a non-Euclidean discrete set of drugs. In contrast, the graph space is over the large graph, and the graph example is only one. Thus, cTS is the only method applicable to graph space other than AdaBatAL.

Extension to constrained cases We simply follow the constrained BO approaches, explained in the section E.4.3; Modelling the probabilistic constraints by GPs and multiplying the probability of constraint satisfaction to the acquisition function.

Training details of AdaBatAL For AdaBatAL, we have two hyperparameters; the number of Nyström samples M, and the tolerance ϵ_{LP}. The number of unlabeled pools N, the batch sizes n, and M need to satisfy the relationship $N \gg M \geq n$. We fixed $M = 500$. As explained in the section F, the larger M yields tighter error bounds for worst-case error but it slows down the computation. We find $M = 500$ works well over the tasks we have tested. For ϵ_{LP}, this is automatically determined for the constrained case via $\epsilon_{\text{LP}} = \epsilon_{\text{vio}}$. For unconstrained cases, we set $\epsilon_{\text{LP}} = 0.01$. For reward function g, we set B-QBC (Riis et al., 2022) for batch AL, and no reward function is set for batch BO. The probabilistic constraints q were modeled by GP as explained. For the intractable expectation of kernel means, we generate $N = 20,000$ data points from the probability distribution μ as explained in section F.

G.2 Baseline Implementations

Table 2 summarizes all baselines. Our method, AdaBatAL, is the only method that can offer adaptive batch size under probabilistic constraints for both AL and BO tasks.

G.2.1 Batch Bayesian Optimization

B3O Budgeted Batch Bayesian Optimization (B3O) (Nguyen et al., 2016) is the only baseline method that offers the adaptive batch size. B3O recasts batch construction as the approximation of acquisition function using a mixture of Gaussians. The adaptive batch size is determined through the marginal likelihood of Gaussian mixture model; the number of Gaussians corresponds to the batch size, and select the batch sizes that yield the largest marginal likelihood, following the standard Bayesian model selection procedure. However, original B3O cannot apply to AL and constrained cases. Simple extension with constraining acquisition function or changing to AL acquisition function could apply to them but we do not investigate in this paper. B3O tends to select around 4-5 batch sizes regardless of the dimension, and is not applicable to large batch size. Moreover, Gaussian mixture model assumption is not always appropriate (e.g. Tanimoto kernel in drug discovery), whereas AdaBatAL naturally adopts these kernel via MMD.

Thompson sampling (TS) Thompson sampling (TS) (Hernández-Lobato et al., 2017) is a random sampling method of $P(x^* \mid \mathbf{D}_t)$ by maximising the function samples drawing from the predictive posterior. Due to its

Table 2: Summary of baseline method. cBO refers to constrained BO.

method	task	adaptive?	constraints?	discrete?	large batch?	any kernel?	any AF?
random	any			✓	✓	✓	✓
AdaBatAL (ours)	any	✓	✓	✓	✓	✓	✓
B3O	BO	✓				✓	✓
TS	BO			✓	✓	✓	
hallucination	BO			✓		✓	✓
LP	BO				✓		✓
TurBO	BO				✓		✓
SOBER	BO			✓	✓	✓	✓
MaxEnt	AL			✓	✓	✓	
BALD	AL			✓	✓	✓	
B-QBC	AL			✓	✓	✓	
ACS-FW	AL			✓	✓	✓	✓
cEI	cBO		✓		✓	✓	
cTS	cBO		✓	✓	✓	✓	
SCBO	cBO		✓		✓		
PropertyDAG	cBO		✓		✓		
PESC	cBO		✓	✓		✓	

random sampling nature, exactly maximising the function samples is not strict when compared to others (e.g. hallucination). Thus, in practice, TS is typically done by taking argmax of function samples amongst the candidates of random samples over input space. This two-step sampling nature (random samples over input space → subsamples with argmax of random function samples) allows us for domain-agnostic BO. However, this scheme itself is a type of acquisition function, so other acquisition function is not naïvely supported. Moreover, due to the random sampling nature, the selected batch samples are not sparsified to efficiently explore uncertain regions.

Hallucination Hallucination (Azimi et al., 2010) tackled batch BO by simulating a sequential process by putting 'fantasy' oracles estimated by GP, translating batch selection into a sequential problem. Hallucination is successful in low batch size n, but not scalable. Even a single iteration of acquisition function maximisation is not trivial due to non-convexity, but they repeat this over n times and produce prohibitive overhead. For discrete and mixed space, maximizing the acquisition function requires enumerating all possible candidates. However, the higher the dimension and larger the number of categorical classes, the more infeasibly large the combination becomes (combinatorial explosion).

Local penalisation (LP) Local penalisation (González et al., 2016), simulates only acquisition function shape change, without fantasy oracles, by penalising acquisition function assuming Lipschitz continuity. This succeeds in speeding up the hallucination algorithm. However, the principled limitations are inherited (combinatorial explosion). Large batch sizes are also not applicable because maximising acquisition function still produces large overhead. This is because maximising acquisition function is typically computed by a multi-start optimiser, but the number of random seeds needs to increase dependent on the number of dimensions and multimodality of the true function. This optimiser also does not guarantee to be globally maximised, which contradicts the assumption of acquisition function (only optimal if it is globally maximised.). Furthermore, Lipschitz continuity assumption limits its applicable range to be only for continuous space.

TurBO TurBO (Eriksson et al., 2019) introduced multiple local BO bounded with trust regions, and allocates batching budgets based on TS. This succeeded in scalable batching via maintaining local BOs that are compact, via shrinking trust regions, based on heuristics with many hyperparameters. Selecting hyperpameters is non-trivial and TurBO cannot apply to discrete and non-Euclidean space, for which kernels do not have lengthscale hyperparameters for the trust region update heuristic (e.g. Tanimoto kernel for drug discovery (Ralaivola et al., 2005)).

SOBER SOBER (Adachi et al., 2022) first introduced the idea of batch BO as a kernel quadrature. Our AdaBatAL is based on SOBER when applying to batch BO tasks. The details are delineated in the section E.2. However, original SOBER is not capable of adaptive batch size or constrained cases.

G.2.2 Constrained Batch Bayesian Optimization

Constrained Expected Improvement (cEI) Constrained expected improvement (cEI) (Letham et al., 2019) is the method based on constrained expected improvement acquisition function (Jones et al., 1998). cEI simply multiplies the probability of constraint satisfaction q_ℓ to the acquisition function. We adopted the official implementation on BoTorch (Balandat et al., 2020). The batching algorithm is based on sample average approximation, a standard batching methid in BoTorch library (Balandat et al., 2020).

Predictive Entropy Search with Constraints (PESC) Predictive Entropy Seach with Constraints (PESC) (Hernández-Lobato et al., 2015) is the constrained version of the predictive entropy search acquisition function (Hernández-Lobato et al., 2014). The official implementation in Spearmint is dependent on Python 2 and is no longer supported in 2023. Thus, we adopted the implementation on BoTorch (Balandat et al., 2020). The batching algorithm is based on Monte Carlo sampling following the original code. However, this code is tremendously slow, which is repeatedly pointed out in BO literature (Eriksson & Poloczek, 2021). We set 7 days as the practical limit of execution time allowing for active learning, and PESC exceeds this limit for almost all tasks except for Hartmann synthetic function. Thus, we only compare PESC on Hartmann task but it was not the best performer.

Scalable Constrained Bayesian Optimization Scalable Constrained Bayesian Optimization (SCBO) is the constrained version of TurBO based on the TS acquistion function and trust region methods. We adopted the official implementation on BoTorch (Balandat et al., 2020) and the same hyperparameters in the original papers (Eriksson et al., 2019) for trust region update heuristics.

Constrained Thompson sampling (cTS) Constrained Thompson sampling (cTS) is the constrained TS method. cTS has not been considered in existing work but this is a simple modification of SCBO. We adopted the two-step sampling used in SCBO for TS and removed the trust region heuristics because this cannot apply to a non-Euclidean kernel (e.g. Tanimoto kernel does not have lengthscale hyperparameter). This is coded based on SCBO implementation on BoTorch (Balandat et al., 2020).

PropertyDAG PropertyDAG Park et al. (2022) is the method based on qNEHVI acquisition function and (Daulton et al., 2020, 2021) for multi-objective optimization. This method assumes (1) ordered constraints but the constraint function is given, (2) multi-objective BO. So it cannot simply apply to our setting as it is. This method is the only one considering ordered case, so we dismantle the components of PropertyDAG to compare in the blackbox ordered constraint case. PropertyDAG consists of three parts: (A) explicit modelling of DAG network in surrogate model (Astudillo & Frazier, 2021), (B) zero inflation model to encode ordered constraint information to qNEHVI acquisition function, and (C) resampling of posterior function samples using sample average approximation to be more likely to satisfy the constraint. We cannot apply (A) and (B) for black-box ordered constraint, because (A) is only for white-box ordered constraint (we cannot model of unknown DAG), and (B) is only for multi-objective BO and specific acquisition function. Thus, we extracted the last part, (C) resampling with sample average approximation, and combined this with cEI, which we refer to PropertyDAG in this paper. We can say this as just resampled version of cEI. The implementation is based on cEI implementation on BoTorch (Balandat et al., 2020) and added the resampling part.

G.2.3 Batch Active Learning

Maximum entropy (MaxEnt) Maximum entropy (MaxEnt) (MacKay, 1992) is the classic acquisition function to select the next query with the largest Shannon entropy. As Riis et al. (2022) pointed out, MaxEnt in FBGP is proportional to the posterior predictive variance. We adopted the following formulation (Riis et al., 2022):

$$\text{MaxEnt} := \mathbb{H}\left[\int \mathbb{P}(y \mid x, \theta) \mathrm{d}\mathbb{P}(\theta \mid \mathcal{D}_0)\right] \propto \mathbb{E}_{\mathbb{P}(\theta|\mathcal{D}_0)}\left[C(x, x \mid \theta)\right] \tag{28}$$

For the batch construction, we take the top n samples following the common practice in batch AL community (Kirsch et al., 2019).

Bayesian Active Learning by Disagreement (BALD) Bayesian active learning by disagreement (BALD) (Houlsby et al., 2011) is another popular objective in Bayesian active learning, is to maximize the expected decrease in posterior entropy (Guestrin et al., 2005). Houlsby et al. (2011) recast the objective from computing entropies in the parameter space to the output space by observing that it is equivalent to maximizing the conditional mutual information between the model's parameters θ and output $\mathbb{I}[\theta, y \mid x, \mathcal{D}_0]$:

$$\text{BALD} := \mathbb{H}\left[\mathbb{E}_{\mathbb{P}(\theta|\mathcal{D}_0)}\left[y \mid x, \mathcal{D}_0, \theta\right]\right] - \mathbb{E}_{\mathbb{P}(\theta|\mathcal{D}_0)}\left[\mathbb{H}\left[y \mid x, \theta\right]\right] \tag{29}$$

Kirsch et al. (2019) pointed out the original BALD criterion is independent selection of a batch of data points leads to data inefficiency as correlations between data points in an acquisition batch are not taken into account. Instead, BatchBALD is proposed whereby we jointly score points by estimating the mutual information between a joint of multiple data points and the model parameters:

$$\text{batchBALD} := \mathbb{H}\left[\mathbb{E}_{\mathbb{P}(\theta|\mathcal{D}_0)}\left[y_1, \dots, y_n \mid x_1, \dots, x_n, \mathcal{D}_0, \theta\right]\right] - \mathbb{E}_{\mathbb{P}(\theta|\mathcal{D}_0)}\left[\mathbb{H}\left[y_1, \dots, y_n \mid x_1, \dots, x_n, \theta\right]\right] \tag{30}$$

We adopted BatchBALD formulation for batch construction.

Bayesian Query-by-Committee (B-QBC) Richardson et al. (2017) propose a Bayesian version of the Query-by-Committee (Seung et al., 1992), using the MCMC samples of the hyperparameters' joint posterior. We query a new data point where the mean predictions $m(x \mid \theta)$ disagree the most. Each mean predictor $m(\cdot \mid \theta)$ drawn from the posterior is equivalent to a single model, and thus this criteria can be seen as a Bayesian variant of a Query-by-Committee, and thus denoted as Bayesian Query-by-Committee (B-QBC). Given that $\bar{m}(x)$ is the average mean function, B-QBC is given as:

$$\text{B-QBC} := \mathbb{V}_{\mathbb{P}(\theta|\mathcal{D}_0)}\left[m(x \mid \theta)\right] = \mathbb{E}_{\mathbb{P}(\theta|\mathcal{D}_0)}\left[\left(m(x \mid \theta) - \bar{m}(x)\right)^2\right] \tag{31}$$

For the batch construction, we take the top n samples following the common practice in batch AL community (Kirsch et al., 2019).

Active Bayesian CoreSets with Frank-Wolfe optimization (ACS-FW) Active Bayesian CoreSets with Frank-Wolfe optimization (ACS-FW) recasts the batch construction as the Bayesian coreset task. Our AdaBatAL is based on ACS-FW when applying to batch AL tasks. The details are deliniated in the section D.1. However, original ACS-FW is not capable of adaptive batch size nor constrained cases. Also, the Bayesian coreset formulation fails to incorporate the predictive uncertainty for batch construction unlike the kernel quadrature formulation. We implemented ACS-FW via following the official code https://github.com/rpinsler/active-bayesian-coresets.

G.3 Dataset

All datasets and tasks are summarized in Table 3.

G.3.1 Synthetic Functions

Hartmann Hartmann 6-dimensional function is defined as:

$$f(x) := -\sum_{i=1}^{4} \alpha_i \exp\left(-\sum_{j=1}^{6} A_{ij}(x_j - P_{ij})^2\right), \tag{32}$$

$$\alpha = (1.0, 1.2, 3.0, 3.2)^\top, \tag{33}$$

$$\mathbf{A} = \begin{pmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{pmatrix}, \tag{34}$$

$$\mathbf{P} = \begin{pmatrix} 1312 & 1696 & 5569 & 124 & 8283 & 5886 \\ 2329 & 4135 & 8307 & 3736 & 1004 & 9991 \\ 2348 & 1451 & 3522 & 2883 & 3047 & 6650 \\ 4047 & 8828 & 8732 & 5743 & 1091 & 381 \end{pmatrix} \tag{35}$$

Table 3: Summary of tasks.

task	method	real/synthetic	space $\mathcal{X}$	dimension	constraints	batch size	kernel
Hartmann	BO	synthetic	continuous	6	-	5-90	RBF
Branin	cBO	synthetic	continuous	2	2	20	RBF
Hartmann	AL	synthetic	continuous	6	-	10	RBF
Ishigami	AL	synthetic	continuous	3	-	10	RBF
Friedman	AL	synthetic	continuous	5	-	10	RBF
Electrolyte	cAL	real-world	continuous	3	2	10	RBF
Cantilever	cAL	real-world	continuous	4	2	10	RBF
Steel	cAL	real-world	continuous	9	1	10	RBF
Ackley	cBO	synthetic	mixed	23	2	200	RBF
Hartmann	cBO	synthetic	continuous	6	2	5	RBF
PestControl	cBO	real-world	discrete	15	2	200	RBF
Malaria	cBO	real-world	discrete	molecule	4	100	Tanimoto
FindFixer	cBO	real-world	graph	node	3	100	graph diffusion
TeamOpt	cBO	real-world	graph	subgraph	3	100	graph diffusion

We take the negative Hartmann function as the objective of BO to make this optimisation problem maximisation. All input variables are continuous with bounds $[0, 1]^6$. The batch size n is 100. The continuous prior is the uniform distribution ranging from [0, 1], following Adachi et al. (2023a). The noisy output is generated by adding i.i.d. zero-mean Gaussian noise with the 0.0192^2 variance to the noiseless $f(x)$.

For constrained BO, we added two constraints; (1) $\sum_{i=1}^d x_i \geq 0.15$ and (2) $\sum_{i=1}^d x_i \leq 3$.

Branin Branin function is defined as:

$$f(x) := \prod_{i=1}^{d} \frac{\sqrt{\sin(x) + 0.5\cos(3x)}}{\sqrt{0.5x} + 0.3}, \tag{36}$$

where the dimension $d = 2$. All input variables are continuous with bounds $x \in [-2, 3]^d$. The batch size n is 20. The continuous prior is the uniform distribution. The noisy output is generated by adding i.i.d. zero-mean Gaussian noise with the 0.0192^2 variance to the noiseless $f(x)$.

For constrained BO, we added two constraints; (1) $\sum_{i=1}^d x_i^2 \leq 4$ and (2) $\sum_{i=1}^d x_i \leq 0$.

Ishigami Ishigami function is defined as:

$$f(x) := \sin(x_1) + 7\sin^2(x_2) + 0.1x_3^4 \sin(x_1), \tag{37}$$

where x_i is the i-th dimensional input and the dimension $d = 3$. All input variables are continuous with bounds $x \in [-\pi, \pi]^d$. The batch size n is 10. The continuous prior is the uniform distribution. The noisy output is generated by adding i.i.d. zero-mean Gaussian noise with the 0.187^2 variance to the noiseless $f(x)$.

Friedman Friedman function is defined as:

$$f(x) := 10\sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5, \tag{38}$$

where x_i is the i-th dimensional input and the dimension $d = 5$. All input variables are continuous with bounds $x \in [0, 1]^d$. The batch size n is 10. The continuous prior is the uniform distribution. The noisy output is generated by adding i.i.d. zero-mean Gaussian noise with the 0.05^2 variance to the noiseless $f(x)$.

Ackely Ackley funciton is defined as:

$$f(x) := -a\exp\left[-b\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right] - \exp\left[\frac{1}{d}\sum_{i=1}^{d}\cos(cx_i)\right] + a + \exp(1) \tag{39}$$

where $a = 20, c = 2\pi, d = 23$. We take the negative Ackley function as the objective of BO to make this optimisation problem maximisation. We modified the original Ackley function into a 23-dimensional function with the mixed spaces of 3 continuous and 20 binary inputs from $[0, 1]^{20}$, following Adachi et al. (2023a). The batch size n is 200. The continuous prior is the uniform distribution ranging from [-1, 1]. The binary prior is the Bernoulli distribution with unbiased weights of 0.5. We assume each of the continuous and binary priors at each dimension is independent.

For constrained BO, we added the two constraints; (1) $x_1 \geq 0$ and (2) $x_2 \geq 0$, where x_1 and x_2 are the first and second dimensions of continuous inputs.

G.3.2 Real-World Functions

Electrolyte Electrolyte is the new problem for the AL task. This is the task of creating the model that predicts the ionic conductivity for the given composition of liquid electrolyte material for the next generation of lithium-ion batteries. This ionic-conductivity function is used for the control model of batteries and plays a crucial role in the control accuracy. However, common practice is to use the lookup table with massive data or pairwise linear function fitting. Collecting the ionic conductivity data requires costly laboratory experiments and fewer data points can accelerate this process while minimizing the cost. GP and AL are powerful frameworks to offer more accurate models with fewer data sizes and cheap models allowing them to be implemented in the control chip. Still, this data collection is under an unknown constraint; the freezing point. While low-temperature operation performance is the key performance indicator of batteries, it causes freezing electrolytes and cannot measure ionic conductivity. The freezing point is dependent on both lithium salt molarity and the cosolvent composition. They show the complex non-linear relationship due to the solvation effect and cannot predict even with the state-of-the-art quantum chemistry simulator. Thus, it is natural to assume this freezing point is an unknown constraint. We create the true function by fitting the experimental data of MA-DMC-EMC-LiPF$_6$ (Logan et al., 2018) system using the Casteel-Amis equation (Casteel & Amis, 1972). Note that Casteel-Amis equation is just for the interpolation of experimental data to be continuous, and is not capable of predicting different cosolvent nor freezing points.

Electrolyte is a three-dimensional continuous input function with two constraints. The input features are (1) the lithium salt (LiPF$_6$) molarity, (2) DMC/EMC cosolvent ratio, and (3) MA/carbonates cosolvent ratio, respectively. The inputs are bounded with $x_1 \in [0, 2]$, $x_2 \in [0, 1]$, and $x_3 \in [0, 0.3]$. The constraints are $x_1 > 0.3$ and $x_2 < 0.9$. The noisy output is generated by adding i.i.d. zero-mean Gaussian noise with the 3^2 variance to the noiseless $f(x)$.

Cantilever Cantilever (Wu et al., 2001) has been proposed for a task to develop a probability-based design optimization framework for ensuring high reliability and safety. This task is to design a cantilever beam under the two failure modes as safety constraints. The objective function to model with GP is the tip displacement, modelled as:

$$f(x) := \frac{4 \times 100^3}{E} \sqrt{X^2 + Y^2}, \tag{40}$$

$$\text{subject to:} \tag{41}$$

$$\frac{f(x) - 4400}{3100} < 2.2535, \tag{42}$$

$$0.8(X + Y) < R. \tag{43}$$

Cantilever is a four-dimensional continuous input function with two constraints. The input features are (1) the yield stress R, (2) the Young's modulus of beam material E, (3) the horizontal load X, and (4) the vertical load Y, respectively. The inputs are bounded with $R \in [3E + 4, 5E + 4]$, $E \in [1E + 7, 5E + 7]$, $X \in [1E + 2, 1E + 3]$, and $Y \in [5E + 3, 5E + 4]$. The noisy output is generated by adding i.i.d. zero-mean Gaussian noise with the 1^2 variance to the noiseless $f(x)$.

Steel Steel (Kuschel & Rackwitz, 1997) has been proposed for design optimization to balance the reliability and cost. This task is to design a steel column under cost constraints. The objective function to model with GP is the limit state function, modelled as:

$$f(x) := F_s - P\left[\frac{1}{2BD} + \frac{F_0 E_b}{BDH(E_b - P)}\right], \tag{44}$$

$$\text{subject to:} \tag{45}$$

$$BD + 5H < 9000, \tag{46}$$

where

$$P := P_1 + P_2 + P_3 \tag{47}$$

$$E_b := \frac{\pi^2 EBDH^2}{2L^2} \tag{48}$$

Steel is the nine-dimensional continuous input function with one constraint. The input features are (1) the yield stress F_s, (2) the dead weight load P_1, (3) the variable load P_2, (4) the variable load P_3, (5) the flange breadth B, (6) the flange thickness D, (7) the profile height H, (8) the initial deflection F_0, and (9) Young's modulus E, respectively. The inputs are bounded with $F_s \in [300, 500]$, $P_1 \in [1E+4, 1E+5]$, $P_2 \in [4E+5, 1E+6]$, $P_3 \in [4E+5, 1E+6]$, $B \in [290, 310]$, $D \in [14, 26]$, $H \in [290, 310]$, $F_0 \in 209800, 210100]$. The noisy output is generated by adding i.i.d. zero-mean Gaussian noise with the 1^2 variance to the noiseless $f(x)$.

PestControl Pest Control (PestControl in the main) is proposed in Oh et al. (2019), which is a multi-categorical optimisation problem (15 dimensions, 5 categories for each dimension). We wish to optimise the effectiveness of pesticides by choosing the 5 actions (selection of pesticides from 4 different firms, or not using any of them), but penalised by their prices. This choice is a sequential decision of 15 stages, and the objective function is expressed as the cumulative loss function with the total of both cost and the portion having pest. The batch size n is 200. We set the categorical prior with equal weights for each choice (discrete uniform distribution). Code is used in https://github.com/xingchenwan/Casmopolitan (Wan et al., 2021).

We added 2 constraints for a more realistic situation. The first constraint is ecosystem change, which assumes exterminating pests too much causes other harmful pests/animals to increase when they reach the hidden threshold. The portion of the product having pests follows the dynamics below:

$$z_i = \alpha_i(1 - x_i)(1 - z_{i-1}) + (1 - \Gamma_i x_i) z_{i-1}, \tag{49}$$

$$z_i \geq z_{\text{limit}}, \tag{50}$$

where i is the number of pest control cycles (15 in total), z is the portion of the product having pest, x is the effectiveness of pesticide that follows a beta distribution with the parameters, which has been adjusted according to the sequence of actions taken in previous control points, α is the action taken (selection of pesticides from 4 different firms, or not using any of it), and z_{limit} is the threshold for ecosystem change (we set $1e-3$). Eq. 50 is the constraint of ecosystem change, and we assume the latent variable z_i is observable.

The second constraint is neighbour disputes, which assume some of the pesticides have unfavourable smells. Neighbours objection follows the Bernoulli distribution and its weights based on the proportion of certain pesticide types and random Gaussian noise. Thus, the feedback to this constraint is in the noisy binary value. If the neighbours' objection is larger than supportive opinion $\theta_{\text{pest}} > 0.5$, a decision maker stops spraying pesticides, thus, objective value cannot be evaluated.

Malaria The objective is to discover an anti-malarial drug exhibiting the smallest EC50 value, which is defined as the concentration of the drug that gives half the maximal response. The lower the concentration, the more effective (better) the drug. The dataset consists of 20,746 small molecules taken from the P. falciparum whole-cell screening derived by the Novatis-GNF Malaria Box (Spangenberg et al., 2013). The molecules are represented as SMILES string and are converted into 2048-dimensional binary features for the Tanimoto kernel. We set four safety constraints, all of which are rules of thumb for judging molecules likely to be oral drugs, shared in drug discovery community (Lipinski et al., 1997; Veber et al., 2002; Butler, 2004; Mochizuki et al., 2019).

The first is Lipinski's rule of five (Lipinski et al., 1997), (A) no more than 5 hydrogen bond donors, (B) no more than 10 hydrogen bond acceptors, (C) A molecular mass less than 500 daltons, (D) A calculated octanol-water partition coefficient that does not exceed 5, (E) no more than 5 rotatable bonds. The second is the Veber filter

(Veber et al., 2002), (A) no more than 10 rotatable bonds, (B) a polar surface area that does not exceed 140. The third is the REOS filter (Butler, 2004), (A) A molecular mass more than 200 daltons and less than 500 daltons, (B) A calculated octanol-water partition coefficient that exceeds -5 but does not exceed 5, (C) no more than 5 hydrogen bond donors, (D) no more than 10 hydrogen bond acceptors, (E) no more than 8 roratable bonds, (F) more than 15 but less than 50 heavy atoms, (G) more than -2 but less than 2 formal charge. The fourth is the drug likeliness filter, (A) A molecular mass less than 400 daltons, (B) at least one ring structure, (C) no more than 5 roratable bonds, (D) no more than 5 hydrogen bond donors, (E) no more than 10 hydrogen bond acceptors, (F) A calculated octanol-water partition coefficient that does not exceed 5.

FindFixer This task is to find the fixer connecting influencers rather than finding the most popular influencer on the social networks graph. A job seeker who wishes to be a celebrity explores the fixer to ask introductions based on graph data using the centrality analysis. Finding a node requires searching on a website or meeting in person, both of which are expensive to evaluate. Fixer can be interpreted as a node with maximum eigenvector centrality under constraints on the degree centrality that does not exceed the threshold (Kiss & Bichler, 2008). In other words, finding the node that is connected to the largest number of nodes with many edges but does not have many edges itself. A job seeker wishes to find the fixer who connects influencers with similar popularity (degree centrality). Thus, the node is constrained based on the degree centrality, and other hidden preference factors. A job seeker judges constraints as a binary value, and the judgment is possibly shaky. We assume the domain is defined as a social network graph synthesized by the Barábsi–Albert model (BA) (Barabási & Albert, 1999).

TeamOpt This task is to organise a team consisting of the most diverse skill sets of members (Wan et al., 2023). The objective is measured by the entropy of the skills of members, assuming the optimal team is when each member is specialised in one skill, and the whole skill distribution is close to uniform. Such teams are positioned on the node of the supergraph, of which edge is the similarity between teams defined as the Jaccord index. The constraints are interpersonal relationships. Every combination of two individuals from N candidates has unobservable hidden continuous likability from 0 to 1. The first is the mean likability constraint, which is the mean of likeability between all possible combinations of members that should be larger than equal-chance. The second is the tragedy-avoidance constraint, which is a binary judge that none of them has a likability lower than a threshold. The third is a flat-relationship constraint, which assumes an entropy of likability must be higher than a threshold. As likability is unobservable, a decision-maker needs to seek advice from many colleagues who partially know each constraint but are noisy estimations.

G.4 Complexity Analysis

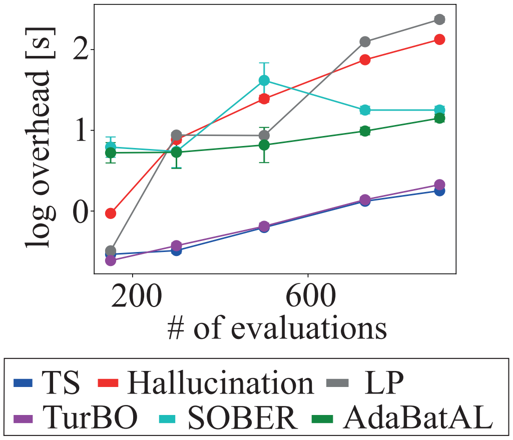


Figure 6: Overhead on Batch Bayesian optimization task for Hartmann ($d = 6$)

As explained in the section 3.8, the time complexity of the AdaBatAL is lower than $\mathcal{O}(NM + M^2 \log n + Mn^2 \log(N/n))$ (Hayakawa et al., 2022), where N is the number of unlabelled pool, M is the number of Nyström samples, and n is the upper bound of the batch size. The space complexity is $\mathcal{O}(NM)$.

We empirically compare the time complexity against the baselines using the Hartmann function with unconstrained batch BO tasks. Figure 6 shows the log overhead to generate the batch samples with different batch sizes that are the same with Figure 3 setting. While TurBO and TS were faster than others, our AdaBatAL was relatively faster than other baselines (SOBER, hallucination, and LP).