GENERATIVE BAYESIAN OPTIMIZATION: GENERATIVE MODELS AS ACQUISITION FUNCTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

We present a general strategy for turning generative models into candidate solution samplers for batch Bayesian optimization (BO). The use of generative models for BO enables large batch scaling as generative sampling, optimization of non-continuous design spaces, and high-dimensional and combinatorial design. Inspired by the success of direct preference optimization (DPO), we show that one can train a generative model with noisy, simple utility values directly computed from observations to then form proposal distributions whose densities are proportional to the expected utility, i.e., BO's acquisition function values. Furthermore, this approach is generalizable beyond preference-based feedback to general types of reward signals and loss functions. This perspective avoids the construction of surrogate (regression or classification) models, common in previous methods that have used generative models for black-box optimization. Theoretically, we show that the generative models within the BO process approximately follow a sequence of distributions which asymptotically concentrate at the global optima under certain conditions. We also demonstrate this effect through experiments on challenging optimization problems involving large batches in high dimensions.

1 INTRODUCTION

Bayesian optimization (BO) has been a successful approach to solve complex black-box optimization problems by making use of probabilistic surrogate models, such as a Gaussian processes (GPs) (Rasmussen & Williams, 2006), and their uncertainty estimates (Shahriari et al., 2016; Garnett, 2023). BO methods have been particularly useful in areas such as hyper-parameter tuning for machine learning algorithms (Snoek et al., 2012), material design (Frazier & Wang, 2016), and robot locomotion (Calandra et al., 2016). The core idea of BO is to apply a Bayesian decision-theoretic framework to make optimal choices by maximizing an expected utility criterion, also known as an acquisition function. The corresponding expectations are taken under a Bayesian posterior over the underlying objective function. Thus, the Bayesian model provides a principled way to account for the uncertainty inherent to the limited amount of data and the noisy observations.

In many applications such as simulated scenarios (Azimi et al., 2010), one is able to run multiple evaluations of the objective function in parallel, even though the simulations themselves might be expensive to run. Common BO approaches to these batch settings incrementally build a set of candidates by sampling "fantasy" observations from the probabilistic model and conditioning on them before selecting the next candidate in the batch (Wilson et al., 2018). Although near-optimal batches can be selected this way, this approach is not scalable to very large batches in high-dimensional spaces, such as problems in protein design (Stanton et al., 2022; Gruver et al., 2023).

One of the most promising alternatives to batch BO has been to train a generative model as a proposal distribution informed by the acquisition function and then sample a batch from the learned proposal (Brookes et al., 2019; Stanton et al., 2022; Gruver et al., 2023; Steinberg et al., 2025). This approach comes with several advantages. Firstly, given a trained generative model, sampling is usually inexpensive. Secondly, existing general-purpose generative models can be used and fine-tuned for the optimization task at hand. Lastly, sampling avoids estimating the global optimum of an acquisition function, which can be hard. However, existing generative approaches to black-box optimization usually rely on fitting a surrogate (regression or classification) model first then training a generative model on top of it (Stanton et al., 2022; Gruver et al., 2023; Steinberg et al., 2025). This two-stage process compounds approximation errors from both models and can increase the computational cost significantly when compared to having a single model.

In this paper we present a general framework for learning generative models for batch Bayesian optimization tasks that requires a single model without the need for additional probabilistic regression or classification surrogates. Our approach for generative BO (GenBO) encodes general utility functions into training objectives for generative models directly. We focus on two cases, one where we train the model via a loss function for a reward model analogously to the direct preference optimization (DPO) formulation for large language models (Rafailov et al., 2023), and the second one where we train the generative model through divergence minimization, using utilities as part of sample weights. We present theoretical analyses on the convergence of approximations and empirical results on practical applications involving high-dimensional combinatorial optimization problems.

2 Background

We consider the problem of estimating the global optimum of an objective function $f : \mathcal{X} \to \mathbb{R}$ as:

$$\mathbf{x}^* \in \operatorname*{argmax}_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) \,, \tag{1}$$

where f is an expensive-to-evaluate black-box function, i.e., $\nabla_{\mathbf{x}} f$ is unavailable. We can only observe $f(\mathbf{x})$ via noisy evaluations $y = f(\mathbf{x}) + \epsilon$, where ϵ is assumed sub-Gaussian (Pisier, 2016). We assume the objective f can be evaluated in parallel, and the algorithm is allowed to run up to $T \geq 1$ optimization rounds with a batch of B query locations $\mathcal{B}_t := \{\mathbf{x}_{t,i}\}_{i=1}^B \subset \mathcal{X}$ per round.

BO with regression models. Typically BO assumes a Bayesian prior over f (Garnett, 2023), often given by a Gaussian process (Rasmussen & Williams, 2006). Given a set of observations $\mathcal{D}_t$, corrupted by Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$, the Bayesian posterior distribution over f given the data $\mathcal{D}_t$ is available in closed form as a GP with known mean and covariance functions (see Appendix A). BO then uses the model's posterior distribution to compute an acquisition function $a_t(\mathbf{x})$ mapping candidate points $\mathbf{x} \in \mathcal{X}$ to their expected utility value $\mathbb{E}[u(y)|\mathbf{x}, \mathcal{D}_t]$, where the utility function u intuitively encodes how useful it is to collect a new observation at $\mathbf{x}$. Classical examples of expected utilities include the probability of improvement $a_t(\mathbf{x}) := p(y \geq \tau|\mathbf{x}, \mathcal{D}_t) = \mathbb{E}[\mathbb{I}[y \geq \tau]|\mathbf{x}, \mathcal{D}_t]$ and the expected improvement $a_t(\mathbf{x}) := \mathbb{E}[\max\{y - \tau, 0\}|\mathcal{D}_t]$. The next candidate is then chosen as:

$$\mathbf{x}_{t+1} \in \operatorname*{argmax}_{\mathbf{x}\in\mathcal{X}} a_t(\mathbf{x}) \,. \tag{2}$$

Batch BO. This strategy can be extended to the batch setting in a variety of ways (Garnett, 2023, §11.3). For instance, one can select the first batch point $\mathbf{x}_{t,1}$ by maximizing a_t as above, and then select the next candidate as $\mathbf{x}_{t,2} \in \operatorname{argmax}_{\mathbf{x}\in\mathcal{X}} \mathbb{E}[u(y)|\mathbf{x}, \mathcal{D}_t \cup \{\mathbf{x}_{t,1}, \tilde{y}_{t,1}\}]$, where the expectation is over both $\tilde{y}_{t,1} \sim p(y|\mathbf{x}_{t,1}, \mathcal{D}_t)$ and $y \sim p(y|\mathbf{x}, \mathcal{D}_t \cup \{\mathbf{x}_{t,1}, \tilde{y}_{t,1}\})$, and iterate over this process until B candidates have been selected for parallel evaluation. Although near optimal, evaluating this conditional expectation becomes quickly intractable as the batch size grows. Hence, one usually resorts to Monte Carlo approximations (Wilson et al., 2018). Other BO strategies allow for efficient optimization of the batch in parallel, such as information-theoretic acquisition functions (Takeno et al., 2020; Teufel et al., 2024), or even asynchronously (Kandasamy et al., 2018). However, scaling up to large batches in high-dimensional domains, especially involving combinatorial or mixed discrete-continuous search spaces, remains challenging (González-Duque et al., 2024).

Active generation with classification models. Instead of relying on a Bayesian surrogate model for f and then computing an acquisition function a on top of it, one can model a directly, which is the main idea behind likelihood-free BO (Song et al., 2022). On this line, methods like variational search distributions (VSD, Steinberg et al., 2025) and batch BORE (Oliveira et al., 2022) learn a probabilistic classifier $\pi(\mathbf{x}) \approx p(y \geq \tau)$ in the original space, $\mathcal{X}$, based on improvement labels $z := \mathbb{I}[y \geq \tau]$ and then generate batches by approximately sampling B candidates from $p(\mathbf{x}|y \geq \tau)$. The classifier can be learned by, e.g., minimizing the cross-entropy loss:

$$L_n(\pi) := -\sum_{i=1}^n z_i \log \pi(\mathbf{x}_i) + (1 - z_i) \log(1 - \pi(\mathbf{x}_i)) \,. \tag{3}$$

Given a prior p_0 over $\mathcal{X}$ and the classifier π_t that minimizes L_{n_t} over the current $n_t := Bt$ data points in $\mathcal{D}_t$, we can now learn a generative model approximating $p(\mathbf{x}|y \geq \tau, \mathcal{D}_t)$ as:

$$q_t \in \operatorname*{argmax}_q \mathbb{E}_{\mathbf{x}\sim q}[\log \pi_t(\mathbf{x})] - \mathbb{D}_{\mathrm{KL}}(q||p_0)\,, \tag{4}$$

which corresponds to an evidence lower bound treating $\pi_t(\mathbf{x}) \approx p(y \geq \tau|\mathbf{x}, \mathcal{D}_t)$ as a likelihood.

Direct preference optimization. The process above for learning q_t can be likened to the typical fine-tuning of large language models (LLMs) via reinforcement learning with human feedback (RLHF, Bai et al., 2022), which would normally involve training the LLM as an RL agent with a reward model ρ. In practice, we do not directly observe rewards, but have access to user preferences. Given a prompt's context c, corresponding to the RL state, let $\mathbf{x}^+, \mathbf{x}^- \sim q(\mathbf{x}|c)$ denote two answers generated by an LLM q, with $\mathbf{x}^+$ denoting the answer preferred by the user, and $\mathbf{x}^-$ the dispreferred one. Having a dataset $\mathcal{D}_n^+ := \{c_i, \mathbf{x}_i^+, \mathbf{x}_i^-\}_{i=1}^n$, one can then learn a reward function ρ by minimizing the negative log-likelihood under a preference model, such as Bradley & Terry (1952):

$$L_n^+(\rho) := -\mathbb{E}_{(c,\mathbf{x}^+,\mathbf{x}^-)\sim\mathcal{D}_n^+}[\log\sigma(\rho(c,\mathbf{x}^+) - \rho(c,\mathbf{x}^-))]\,. \tag{5}$$

Having learned a reward model ρ_n, RLHF trains the LLM as to approximate an agent's optimal policy under ρ_n. Regularization based on the Kullback-Leibler (KL) divergence with respect to a reference model q_{ref} is further added to improve stability. The optimal generative model then solves:

$$q_n \in \operatorname*{argmax}_q \mathbb{E}_{c\sim\mathcal{D}_n^+,\mathbf{x}\sim q(\mathbf{x}|c)}[\rho_n(c,\mathbf{x})] - \beta\mathbb{D}_{\mathrm{KL}}(q||q_{\mathrm{ref}})\,. \tag{6}$$

Direct preference optimization (DPO, Rafailov et al., 2023) removes the need for an explicit reward model by viewing the LLM itself through the lens of a reward model. It is not hard to show that, fixing a reward model ρ, the optimal solution to Equation 6 is given by:

$$q(\mathbf{x}|c) = \frac{1}{\zeta_\rho(c)} q_{\mathrm{ref}}(\mathbf{x}|c) \exp\left(\frac{1}{\beta}\rho(c,\mathbf{x})\right), \tag{7}$$

where $\zeta_\rho(c) := \sum_{\mathbf{x}} q_{\mathrm{ref}}(\mathbf{x}|c)\exp(\beta^{-1}\rho(c,\mathbf{x}))$ is the partition function at the given context c. Although it is intractable to evaluate ζ_ρ in practice, DPO uses the fact that, in the Bradley-Terry model, the partition function-dependent terms cancel out. Note that the reward model ρ can be expressed in terms of the optimal q as:

$$\rho(c,\mathbf{x}) = \beta\log\left(\frac{q(\mathbf{x}|c)}{q_{\mathrm{ref}}(\mathbf{x}|c)}\right) + \beta\log\zeta_\rho(c)\,. \tag{8}$$

Applying the substitution above to the preference-based loss (5), we get:

$$L_{\mathrm{DPO}}(q) = -\mathbb{E}_{(c,\mathbf{x}^+,\mathbf{x}^-)\sim\mathcal{D}_n^+}\left[\log\sigma\left(\beta\log\left(\frac{q(\mathbf{x}^+|c)}{q_{\mathrm{ref}}(\mathbf{x}^+|c)}\right) - \beta\log\left(\frac{q(\mathbf{x}^-|c)}{q_{\mathrm{ref}}(\mathbf{x}^-|c)}\right)\right)\right], \tag{9}$$

which eliminates the partition function ζ_ρ terms. Therefore, we can train the generative model q directly with L_{DPO} without the need for an intermediate reward model. Such simplification to a single training loop cuts down the need for computational resources, eliminates a source of approximation errors (from learning ρ), and brings in theoretical guarantees from Bradley-Terry models (Shah et al., 2016; Bong & Rinaldo, 2022). The main question guiding our work is whether we can apply a similar technique to simplify the training of (arbitrary, not necessarily LLM) generative models for likelihood-free BO by removing the need for an intermediate surrogate model for f.

3 A GENERAL RECIPE FOR GENERATIVE BAYESIAN OPTIMIZATION

As seen in Section 2, using generative models for BO typically involves training a regression or classification model as an intermediate step to then train the candidate generator. The use of an intermediate model demands additional computational resources and brings in further sources of approximation errors which may hinder performance. Hence, we propose a framework to train the generative model directly from (noisy) observation values. The main idea is to train the model to approximate a target distribution proportional to BO's acquisition function and then use the learned generative model as a proposal for the next query locations. There are different approaches to do so, some of which have been previously explored in the literature, for specific acquisition functions, such as the probability of improvement (Brookes et al., 2019; Steinberg et al., 2025) and upper confidence bound (Yun et al., 2025). However, we here focus on a general recipe to turn a generative model into a density following *any* acquisition function that can be expressed as an expected utility.

Utility functions. Consider a likelihood-free BO setting (Song et al., 2022), where we aim to directly learn an acquisition function $a_t : \mathcal{X} \to \mathbb{R}$ at every BO round $t \in \{1, \dots, T\}$ based on available data. If our acquisition function takes the form of an expected utility:

$$a_t(\mathbf{x}) = \mathbb{E}[u_t(\mathbf{x})|\mathcal{D}_{t-1}], \tag{10}$$

we can estimate it from noisy samples $\{\mathbf{x}_i, u_{t,i}\}_{i=1}^{t-1}$, where $\mathbb{E}[u_{t,i}|\mathcal{D}_{t-1}] = \mathbb{E}[u_t(\mathbf{x}_i)|\mathcal{D}_{t-1}]$. For example, we have:

1. Probability of improvement (PI): $u_{t,i} = \mathbb{I}[y_i \geq \tau_t]$;
2. Expected improvement (EI): $u_{t,i} = \max(y_i - \tau_t, 0)$;
3. Simple regret (SR): $u_{t,i} = y_i$;

given a threshold τ_t for improvement-based utilities, e.g., $\tau_t := \max_{i<t} y_i$ or a quantile of the empirical marginal observations distribution (Tiao et al., 2021). A comprehensive summary of typical utility functions for BO can be found in Wilson et al. (2018). The ones listed above, however, we can write directly as a function of the observations. We also use a soft-plus version of EI (sEI) in our experiments, which remains positive at a low value when $y = \tau$.

BO with generative models. As an example, consider the case of PI where $a(\mathbf{x}) = \mathbb{E}[\mathbb{I}[y \geq \tau]] = p(y \geq \tau|\mathbf{x})$, which has been previously applied to train generative models for black-box optimization via surrogates (Steinberg et al., 2025). Given a sampler for the conditional distribution $p(\mathbf{x}|y \geq \tau)$, by Bayes rule, we recover the original PI as:

$$a(\mathbf{x}) = p(y \geq \tau|\mathbf{x}) = \frac{p(\mathbf{x}|y \geq \tau)p(y \geq \tau)}{p_0(\mathbf{x})} \propto \frac{p(\mathbf{x}|y \geq \tau)}{p_0(\mathbf{x})}\,. \tag{11}$$

As the prior p_0 is usually known, and it can even be set as uninformative $p_0(\mathbf{x}) \propto 1$, we see that learning a generative model to approximate the posterior above is equivalent to learning a probabilistic classifier for the improvement event $y \geq \tau$. Moreover, if we only have a probabilistic classifier approximating $p(y \geq \tau|\mathbf{x})$, we still need to select candidate points via optimization over the classification probabilities landscape, which can be highly non-convex presenting several local optima, recalling that in the usual BO setting we choose $\mathbf{x}_{t+1}$ as the (global) maximizer of the acquisition function a. In contrast, a generative model provides us with a direct way to sample candidates $\mathbf{x} \sim p(\mathbf{x}|y \geq \tau)$ which will by default concentrate at the highest density regions, and consequently highest utility, according to the model. Finally, note that this same reasoning can be extended to any other non-negative expected utility function by training the generative model to approximate:

$$p_t^*(\mathbf{x}) \propto p_0(\mathbf{x})a_t(\mathbf{x})\,, \tag{12}$$

or similarly $p_t^*(\mathbf{x}) \propto p_0(\mathbf{x}) \exp a_t(\mathbf{x})$, which allows for utilities that might take negative values.

Overview. Let $\mathcal{Q} \subset \mathcal{P}(\mathcal{X})$ be a learnable family of probability distributions over a given domain $\mathcal{X}$. We consider general loss functions of the form:

$$L_t(q) := \lambda_t R_t(q) + \sum_{i=1}^{n_t} \ell_i(q)\,, \tag{13}$$

where ℓ_i are individual losses over points $\mathbf{x}_i \in \mathcal{X}$ or pairs of points $\mathbf{x}_{i,1}, \mathbf{x}_{i,2} \in \mathcal{X}$ and their corresponding utility values, $\lambda_t \geq 0$ is an optional regularization factor, and $R_t : \mathcal{Q} \to [0, \infty)$ is a complexity penalty function. The algorithm then proceeds by learning a proposal distribution as:

$$q_t \in \operatorname*{argmin}_{q \in \mathcal{Q}} L_t(q)\,. \tag{14}$$

A batch $\mathcal{B}_{t+1} := \{\mathbf{x}_{t+1,i}\}_{i=1}^{B}$ is sampled from the learned proposal q_t. We evaluate the utilities $u_{t+1}(y_{t+1,i})$ with the collected observations $y_{t,i} \sim p(y|\mathbf{x}_{t+1,i})$, for $i \in \{1, \dots, B\}$, and repeat the cycle up to a given number of iterations $T \in \mathbb{N}$. This process is summarized in Algorithm 1 in the appendix. In the following, we describe approaches to formulate general loss functions for learning acquisition functions and how to ensure that the sequence of batches $\{\mathcal{B}_t\}_{t=1}^{\infty}$ asymptotically concentrates at the optimum $\mathbf{x}^*$.

3.1 Preference-based learning

We aim to apply a similar reparameterization trick to the one in DPO to simplify generative BO methods. Note that, for a general classification loss, such as the one in Equation 3, it is not possible to eliminate the partition function resulting from a DPO-like reparameterization without resorting to approximations, which might change the learned model. Hence, we need a pairwise-contrastive training objective.

Preference loss. To apply a preference-based loss, we can train a model to predict preferential directions of the acquisition function. Assume we have a dataset $\mathcal{D}_n^u := \{\mathbf{x}_i, u_i\}_{i=1}^n$ with n evaluations of a given utility function $u : \mathbb{R} \to \mathbb{R}$. We may reorganize the data into pairs of inputs and corresponding utility values $\{\mathbf{x}_{i,1}, \mathbf{x}_{i,2}, u_{i,1}, u_{i,2}\}_{i=1}^{n/2}$, where $u_{i,j} := u(y_{i,j})$, for $j \in \{1, 2\}$, and train a generative model q using the Bradley-Terry preference loss from DPO with, for $i \in \{1, \dots, n/2\}$:

$$\ell_i^{\mathrm{PL}}(q, \Delta u_i) := -\log \sigma \left(\beta \operatorname{sign}(\Delta u_i) \left(\log \left(\frac{q(\mathbf{x}_{i,1})}{p_0(\mathbf{x}_{i,1})} \right) - \log \left(\frac{q(\mathbf{x}_{i,2})}{p_0(\mathbf{x}_{i,2})} \right) \right) \right), \tag{15}$$

where $\Delta u_i := u_{i,1} - u_{i,2}$, as in the DPO formulation, $\beta > 0$ is a (optional) temperature parameter and the prior p_0 can be given by a reference model, either pre-trained or derived from expert knowledge about feasible solutions to the optimization problem (1). Similar to Rafailov et al. (2023), the learned generative model is seeking to approximate:

$$p_u^*(\mathbf{x}) := \frac{1}{\zeta_u} p_0(\mathbf{x}) \exp \left(\frac{1}{\beta} \mathbb{E}[u(y)|\mathbf{x}] \right), \tag{16}$$

where ζ_u is the normalization factor.

Robust preference loss. As shown in Chowdhury et al. (2024), the original DPO loss is not robust to preference noise. As in BO, one usually only observes noisy evaluations of the objective function, utility values directly derived from the observation values will also be noisy and correspondingly the sign of their differences as well. Namely, assume there is a small $p_{\text{flip}} \in (0, 1/2)$ probability of the preference directions being flipped w.r.t. the sign of the true expected utility:

$$\mathbb{P} \left[\operatorname{sign}(u_{i,1} - u_{i,2}) = \operatorname{sign}(\mathbb{E}[u_{i,2}|\mathbf{x}_{i,2}] - \mathbb{E}[u_{i,1}|\mathbf{x}_{i,1}]) \right] = p_{\text{flip}}\,. \tag{17}$$

Chowdhury et al. (2024) showed that the original DPO preference loss is biased in this noisy case, and proposed a robust version of the DPO loss to address this issue as:

$$\ell_i^{\mathrm{rPL}}(q, \Delta u_i) := \frac{(1 - p_{\text{flip}})\ell_{\mathrm{PL}}(q, \Delta u_i) - p_{\text{flip}} \ell_i^{\mathrm{PL}}(q, -\Delta u_i)}{1 - 2p_{\text{flip}}}, \tag{18}$$

which yields the robust preference loss (rPL): $L_n^{\mathrm{rPL}}(q) := \sum_{i=1}^n \ell_i^{\mathrm{rPL}}(q, \Delta u_i)$. It follows that the loss function above is unbiased and robust to observation noise.

3.2 Divergence-based learning

A disadvantage of DPO-based losses when applied to BO is that they only take the signs of the pairwise utility differences into account, discarding the remaining information contained in the magnitude of the utilities. A simpler approach is to train the generative model q to match p_u^* directly.

Forward KL. If we formulate the target distribution as $p_u^* \propto p_0(\mathbf{x})a(\mathbf{x})$, the forward Kullback-Leibler (KL) divergence of the proposal w.r.t. the target is given by:

$$\mathbb{D}_{\mathrm{KL}}(p_u^* || q) = \mathbb{E}_{\mathbf{x} \sim p_u^*} \left[\log p_u^*(\mathbf{x}) - \log q(\mathbf{x}) \right]. \tag{19}$$

As we do not have samples from p_u^*, at each iteration t the algorithm generates samples from the current best approximation $\mathcal{B}_t := \{\mathbf{x}_{t,i}\}_{i=1}^B \sim q_t$. An unbiased training objective can then be formulated as:

$$\ell_i^{\mathrm{fKL}}(q) = -\frac{p_0(\mathbf{x}_i)}{q_{i-1}(\mathbf{x}_i)} u(y_i) \log q(\mathbf{x}_i)\,, \tag{20}$$

which we write in a condensed form to avoid notation clutter with $q_i = q_{\lfloor i/B \rfloor}$ and n corresponding to the total number of observations up to a given round. The objective above is unbiased and its global optimum can be shown to converge to p_u^* by an application of standard results from the adaptive importance sampling literature (Delyon & Portier, 2018). A simpler version of this training objective was derived for CbAS (Brookes et al., 2019) using only the last batch for training, which would allow for convergence as the batch size goes to infinity $B \to \infty$. Furthermore, as we will see in our analysis, convergence to p_u^* is not sufficient to ensure convergence to the global optima of the objective function f.

Balanced forward KL. As utilities like those of PI and EI can evaluate to 0 at the points where $y < \tau$ was observed, with τ corresponding to an improvement threshold, every point below the threshold will not be penalized by the loss function. As a result, the model may keep high probability densities in regions of low utility. To prevent this, we may use an alternative formulation of the forward KL which comes from the definition of Bregman divergences with the convex function $u \mapsto u \log u$, yielding a loss:

$$\ell_i^{\mathrm{bfKL}}(q) = -\frac{p_0(\mathbf{x}_i)}{q_{i-1}(\mathbf{x}_i)} u(y_i) \log q(\mathbf{x}_i) + \frac{q(\mathbf{x}_i)}{q_{i-1}(\mathbf{x}_i)} . \tag{21}$$

We defer the details of the derivation to the appendix. Although the additional $q(\mathbf{x})$ only contributes to a constant term when integrated over, for finite-sample approximations, it contributes to a soft penalty on points where we observed $u(y) = 0$.

3.3 Generalizations

In general, we can extend the above framework to use proper scoring rule $S : \mathcal{P}(\mathcal{X}) \times \mathcal{X} \to \mathbb{R}$ (Gneiting & Raftery, 2007) other than the log loss. We can then learn q approximating p_u^* by minimizing:

$$L_n^S(q) = -\sum_{i=1}^{n} \frac{p_0(\mathbf{x}_i)}{q_{i-1}(\mathbf{x}_i)} u(y_i) S(q, \mathbf{x}_i) . \tag{22}$$

Although we leave the exploration of this formulation for future work, it is readily extensible to other types of generative models which may not have densities available in closed form, such as diffusion and flow matching (Lipman et al., 2024), which still provide flexible probabilistic models.

4 Theoretical analysis

In this section, we present a theoretical analysis of the algorithm's approximation of the utility-based target distribution and its performance in regards to the global optimization problem (1). We consider parametric generative models q_θ with a given parameter space $\theta \in \Theta \subset \mathbb{R}^M$. For the purpose of our analysis, we will assume that models can be described as $q_\theta(\mathbf{x}) = \exp g_\theta(\mathbf{x})$, which is always possible whenever densities are strictly positive $q_\theta(\mathbf{x}) > 0$. To accommodate for both the pairwise preference-based losses and the point-based divergence approximations, we introduce the following notation for the loss function:

$$L_n(g_\theta) = \bar{R}_n(g_\theta) + \sum_{i=1}^{n} \ell(m_i(g_\theta), z_i) , \tag{23}$$

where $m_i(g_\theta)$ corresponds to the model evaluation at data point i with, e.g., $m_i(g_\theta) := \log q_\theta(\mathbf{x}_i)$ for KL, and $m_i(\theta) := \log q_\theta(\mathbf{x}_{i,1}) - \log q_\theta(\mathbf{x}_{i,2})$ for preference-based losses, and z_i encodes the dependence on utility values with $z_i := u(y_i)$ for KL and $z_i := \mathrm{sign}(u_{i,1} - u_{i,2})$ for DPO losses. We set $\bar{R}_n$ as an extended regularizer $\bar{R}_n(g) := \lambda_n R_n(g) + \frac{\bar{\lambda}_n}{2}(\int_{\mathcal{X}} \exp g(\mathbf{x}) \,\mathrm{d}\mu(\mathbf{x}) - 1)^2$, where μ corresponds to the underlying base measure on the domain $\mathcal{X}$ (i.e., the counting measure for discrete domains or the Lebesgue measure for Euclidean spaces). Note that the additional term is always zero for the generative models, as $\int_{\mathcal{X}} \exp g_\theta(\mathbf{x}) \,\mathrm{d}\mu(\mathbf{x}) = \int_{\mathcal{X}} q_\theta(\mathbf{x}) \,\mathrm{d}\mu(\mathbf{x}) = 1$, but including it here facilitates our analysis to operate with any unconstrained $g : \mathcal{X} \to \mathbb{R}$.

Regularity assumptions. We make a few mild regularity assumptions about the problem setting and the model. Firstly, for the analysis, we assume that both the models g_θ lie in a reproducing kernel Hilbert space (RKHS) $\mathcal{H}_k$ shared with the true log density g_*, which is such that $p_u^*(\mathbf{x}) = \exp g_*(\mathbf{x})$.

The domain $\mathcal{X}$ is assumed to be a compact metric space with main results specialized for the finite discrete setting, i.e., $|\mathcal{X}| < \infty$. The model $q_\theta(\mathbf{x})$ is continuously twice differentiable with respect to the parameters $\theta \in \Theta$ with bounded second-order derivatives. The individual losses $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ are strictly convex and twice differentiable w.r.t. their first argument. In addition, we assume that, at the target g_* the individual loss $\ell(m_i(g_*), z_i)$ is conditionally sub-Gaussian (Boucheron et al., 2013; Chowdhury & Gopalan, 2017) w.r.t. the data-generating process, basically meaning that the probability distribution of each loss has zero mean and light tails. We also assume that the regularizer $R_n : \mathcal{Q} \to [0, \infty)$ is strongly convex and twice differentiable. We defer the formal assumptions and their discussion to the appendix.

Lemma 1. *Let Assumption A1, A2 and A3 be satisfied. Then, for any $g \in \mathcal{H}_k$, the following holds:*

$$\frac{1}{2}\|g - g_n\|^2_{H_n} \leq L_n(g) - L_n(g_n) \leq \frac{1}{2}\|\nabla L_n(g)\|^2_{H_n^{-1}}\,, \tag{24}$$

where $H_n : \mathcal{H}_k \to \mathcal{H}_k$ is an operator-valued lower bound on the Hessian of the loss L_n:

$$\forall g \in \mathcal{H}_k, \quad \nabla^2 L_n(g) \succeq H_n := \lambda I + \alpha_\ell \sum_{i=1}^{n} m_i \otimes m_i\,, \tag{25}$$

where $\phi(\mathbf{x}) := k(\cdot, \mathbf{x})$, for $\mathbf{x} \in \mathcal{X}$.

Remark 1. The result in Lemma 1 automatically ensures that the loss functional L_n is strongly convex, as $\nabla^2 L_n(g) \succeq H_n \succeq \lambda I \succ 0$, for all $g \in \mathcal{H}_k$, and therefore has a unique minimizer at g_n. The same, however, cannot be implied about $L_n(g_\theta)$ over Θ based solely on this result, since the mapping $\theta \mapsto g(\cdot, \theta)$ might be non-linear.

Corollary 1. *Consider the setting in Lemma 1, and assume that there is $\theta_* \in \Theta$ such that $g_{\theta_*} = g_*$. Then, given any $\delta \in (0, 1)$, the following holds with probability at least $1 - \delta$:*

$$\forall n \in \mathbb{N}, \quad |\langle m, g_* \rangle_k - \langle m, g_{\theta_n} \rangle_k| \leq 2\beta_n(\delta)\|m\|_{H_n^{-1}}\ \forall m \in \mathcal{H}_k,$$

where $\beta_n(\delta) := \lambda^{-1/2}\|\nabla \bar{R}_n(g_*)\|_k + \sigma_\ell\sqrt{2\alpha_\ell^{-1}\log(\det(\mathbf{I} + \alpha_\ell \lambda^{-1} M_n^\top M_n)^{1/2}/\delta)}$, and $M_n := [m_1, \dots, m_n]$.

The result above is a direct consequence of Theorem 1 in the appendix, and it shows that the approximation error for the optimal parameter θ_n concentrates similarly to that of a kernel method, even though we do not require the model to be a kernel machine. In addition, the term $\|m\|_{H_n^{-1}}$ is associated with the predictive variance of a Gaussian process model, which can be shown to converge to zero whenever $\inf_{\mathbf{x} \in \mathcal{X}} q_\theta(\mathbf{x}) \geq b_q > 0$, for all $\theta \in \Theta$ (see Lemma 4 in the appendix).

Optimality. Corollary 1 and the latter allows us to establish that the model converges to the target g_* associated with the target distribution p_u^* for a given utility function u. However, convergence to the target distribution alone does not ensure optimality of the samples $\mathbf{x} \sim q_t$. The latter is possible by applying results from reward-weighted regression, which shows that training a proposal to maximize $\mathbb{E}_{y \sim p(y|\mathbf{x}), \mathbf{x} \sim q_{t-1}}[u(y) \log q(\mathbf{x})]$ yields a sequence of increasing expected rewards $\mathbb{E}[u(y_t)] \leq \mathbb{E}[u(y_{t+1})] \leq \dots$ (Štrupl et al., 2022, Thm. 4.1). If the maximizer of the sequence of expected utilities converges to the maximizer of the objective function f, then the generative BO proposals will concentrate at the true optimum $\mathbf{x}_*$. Therefore, for KL-based loss functions, one may drop the proposal densities in the importance sampling weights $1/q_{i-1}(\mathbf{x}_i)$ to promote this posterior concentration phenomenon, as corroborated by our experimental findings, which generally did not include importance weights. This same concentration of the learned target distribution should also occur with the preference-based loss functions due to the absence of importance-sampling weights.

5 Related work

Using generative models for online-optimization is becoming an increasingly popular method for optimization in discrete, mixed discrete-continuous or high dimensional design spaces where classical BO is limited. The following discusses other works applying generative models to BO settings and contrasts them with the reward-model-free active generation framework we propose.

Latent-space BO. In latent-space BO (LSBO) methods for high-dimensional problems (Gómez-Bombarelli et al., 2018; Stanton et al., 2022; Gruver et al., 2023), one learns a probabilistic representation of a (usually lower-dimensional) manifold of the data jointly with f, and performs BO in that space, projecting query points back to the original space at evaluation time. This technique has led to numerous BO methods for high-dimensional and discrete-space optimization (Gómez-Bombarelli et al., 2018; Gruver et al., 2023; González-Duque et al., 2024). Learning this latent-space can, however, cause complications. LSBO can suffer poor sample efficiency if the latent-space is learned from the initial training set and then fixed (Tripp et al., 2020). Or poor performance can arise from reconstruction errors between the latent and observation space (Lee et al., 2025). GenBO and other methods like VSD do not suffer from these issues as all inference is done in the observation space. Despite recent advances in the field (Chu et al., 2024; Lee et al., 2025; Moss et al., 2025), to our knowledge, LaMBO-2 remains state-of-the-art in LSBO for *long* sequences, like proteins.

Diffusion for BBO. There has been recent progress in adapting denoising diffusion models to black-box optimization (BBO) tasks, often by learning a model that can be conditioned on observation values, given a dataset of evaluations (Krishnamoorthy et al., 2023). Other approaches involve guiding the diffusion process by a given utility function derived from a regression model (Gruver et al., 2023; Yun et al., 2025). Note, however, that such methodologies are specific to diffusion, whereas we focus on a general approach that can be applied to arbitrary generative models.

LLMs and BO. Recent work has begun to integrate large language models (LLMs) into BO pipelines, primarily to inject prior knowledge, improve cold-start performance, or offload certain design decisions to a learned policy. Several studies use LLMs as contextual priors over the design space: for example, guiding initialization or proposal generation by leveraging natural-language domain knowledge (Liu et al., 2024), or selecting acquisition functions adaptively via an LLM-driven controller (Aglietti et al., 2025). Other work treats BO as a test-time search tool that an LLM can call to refine or validate its own proposals during inference (Agarwal et al., 2025). Most relevant to our setting is a recent reward-model-free approach for protein engineering (Chen et al., 2025), which uses LLM preference modeling, akin to DPO, to steer search without an explicit surrogate. This shares the reward-model-free philosophy of GenBO, but differs fundamentally in relying on a general-purpose LLM, whereas GenBO provides a framework for task-specific generative black-box optimization problems with no language interface or pretrained reward structure.

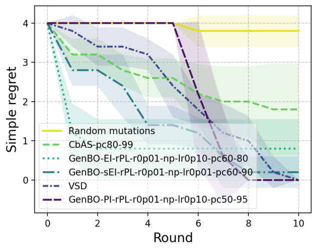
6 Experiments

We evaluate several variants of generative BO (GenBO) on a number of challenging sequence optimization tasks against popular and strong baselines, including CbAS (Brookes et al., 2019), VSD (Steinberg et al., 2025), and LaMBO-2 (Gruver et al., 2023), besides trivial baselines, random mutations and a genetic algorithm (GA) implemented in poli (González-Duque et al., 2024). As performance measures, we assess the simple regret, $r_t := f(\mathbf{x}^*) - \max_{i \leq n_t} f(\mathbf{x}_i)$, and the cumulative maximum, $\max_{i \leq n_t} f(\mathbf{x}_i)$, where $n_t := Bt$ is the number of function evaluations up to round t. In legend boxes, algorithms are sorted in descending order of final average regret. Shaded areas correspond to ± 1 standard deviation across five different random seeds. Appendix D presents further details about experiment settings and ablations. Table 4 and 5 summarize final results.

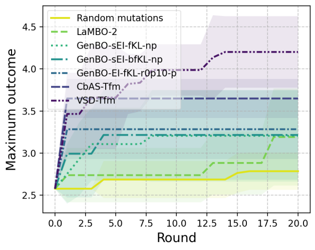
6.1 Text optimization

As a first experiment, we wish to optimize a short sequence (5 letters) to minimize the edit distance to the sequence `ALOHA`, which is implemented as a poli black-box (González-Duque et al., 2024). Here $\mathcal{X} = \mathcal{V}^M$ where $\mathcal{V}$ is the English alphabet, and M is sequence length. Even though this sequence is relatively short, still $|\mathcal{X}| = |\mathcal{V}|^M > 11.8$ million elements. We increase the difficulty by only allowing $|\mathcal{D}_0| = 64$ where the minimum edit distance is 4, $B = 8$, and $T = 10$. We compare GenBO to the classifier guided VSD (Steinberg et al., 2025) and CbAS (Brookes et al., 2019), and to a simple greedy baseline that applies (3) random mutations to its best candidates per-round (González-Duque et al., 2024). For GenBO, VSD and CbAS we use a simple mean-field (independent) categorical proposal distribution, q, and a uniform prior, p_0. VSD and CbAS use a simple embedding and 1-hidden layer MLP classifier for estimating PI. We also varied the threshold τ annealing schedule. Architectural details and other experimental specifics are given in Section D.1.

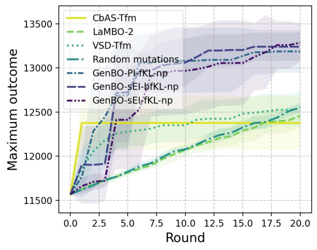
Results are summarized in Figure 1a. We can see that the random baseline is not able to make much headway and CbAS under-performs due to its limited use of data (last batch only) in retraining. For this experiment, GenBO with the robust preference loss (rPL) and EI-based utilities showed the quickest improvements, whereas PI is able to reach the exact optimum at the end, with VSD eventually also achieving good performance. In Figure 5 (appendix), we present an ablation study on the threshold τ_t annealing scheme we used to balance the exploration-exploitation trade-off for GenBO and PI-based baselines (VSD and CbAS). The plots reveal that this problem generally favors a more exploitative approach by concentrating on higher quantiles of the observations marginal distribution. GenBO was, however, relatively less sensitive to the choice of annealing scheme, as long as the final percentile was set anywhere above 90%, whereas VSD required a generally sharper rise to above the 95% quantile towards the end of the optimization process, favoring original settings suggested by Steinberg et al. (2025). We also find that in this problem the use of a pre-trained informative prior p_0 may not bring significant performance advantage, as GenBO variants with no prior (i.e., $p_0 \propto 1$) performed best. Lastly, we also highlight significant improvements in run time for GenBO, making it on average 3 times faster than VSD (see Table 6 in the appendix) for not needing to fit an intermediate surrogate model.



(a) ALOHA

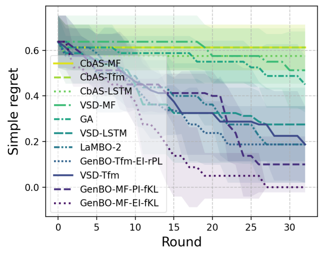


(b) Stability

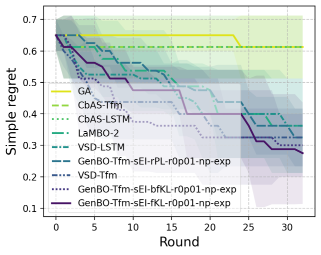


(c) SASA

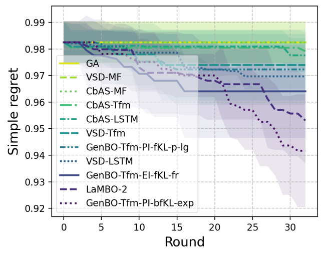
Figure 1: Performance of baseline black box optimizers and GenBO variants on the (a) ALOHA, (b) stability, and (c) solvent accessible surface area optimization problems.



(a) $M = 15$



(b) $M = 32$



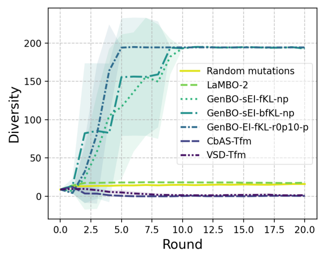
(c) $M = 64$

Figure 2: Simple regret of the baseline black box optimizers and the GenBO variants on the Ehrlich closed-form test function protein design task for varying sequence lengths, M.

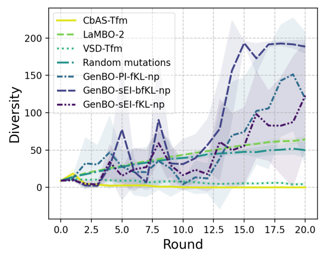
6.2 PROTEIN DESIGN

We now consider three protein sequence design tasks where $|\mathcal{V}| = 20$ and we have varying M. We again use VSD, CbAS and random mutation as baselines, and add to them the guided diffusion based LaMBO-2 (Gruver et al., 2023). GenBO, VSD and CbAS all share the same generative backbone, which is the causal transformer used in Steinberg et al. (2025); VSD and CbAS also use the same CNN-classifier guide used in that work. We present additional architectural information, and additional experimental details in Section D.2. We use the black-box implementations in POLI for these tasks, and POLI-BASELINES implementations of the random and LaMBO-2 baselines.

The first task we consider is optimization of the Ehrlich functions introduced by Stanton et al. (2024). These are challenging biologically inspired parametric closed-form functions that explicitly simulate nonlinear (epistatic) effects of sequence on outcome. The outcomes are $y \in \{-1\} \cup [0, 1]$ where -1 is reserved for infeasible sequences. We use the same protocol as in Steinberg et al. (2025), where we optimize sequences of length $M = \{15, 32, 64\}$ all with motif lengths of 4, and $|\mathcal{D}_0| = 128$, $T = 32$ and $B = 128$. The results are summarized in Figure 2. We again see that GenBO variants are able to outperform or match the performance of baselines, with KL-based losses yielding the best performance. In higher dimensions with the longest sequence setting, the benefits of the balanced forward KL loss, with its density minimization effect in areas of lower utility, are more evident. In addition, we note that exponential regularization, corresponding to assuming an exponential dot-product kernel for the RKHS feature space of the model (see Remark 2), allowed for the best performance in higher dimensions. Lastly, in Figure 6 (appendix), we present an ablation study on the batch size setting B, showing monotonic improvements, especially for large $B \geq 32$.



(a) Stability



(b) SASA

Figure 3: Batch diversity scores per round on the FoldX protein optimization tasks.

For our final set of experiments we present two real protein optimization tasks. These experiments have been adapted from Stanton et al. (2022) where the aims are to maximize the stability and solvent accessible surface (SASA) of the proteins, respectively. The black-box is the FoldX molecular simulation software (Schymkowitz et al., 2005), and is wrapped by POLI (González-Duque et al., 2024). We chose the `mRouge` red fluorescent protein ($M = 228$) as the base protein for the tasks. Both tasks were given $T = 20$ rounds, a batch size of $B = 64$, and an initial training set of $|\mathcal{D}_0| = 88$ as a subset from Stanton et al. (2022). Results are summarized in Figure 1b for stability and Figure 1c for SASA. All variants of GenBO find the stability task challenging, along with the LaMBO-2 and random baselines. CbAS and especially VSD are better able to stabilize this protein. As shown by diversity scores in Figure 3a, which we measure by averaging the Levenshtein distance across the batch in the same way as Steinberg et al. (2025), algorithmic baselines with the lowest diversity yielded top performance, indicating that pure exploitation from around the starting dataset led to the highest outcomes. However, most variants of GenBO far outperform the baselines on the SASA task, and much more rapidly. We believe this task favors extrapolation away from the prior, due to the high performance of GenBO variants with uninformative prior. In contrast to the stability, the diversity scores show that increasing exploration led to better outcomes for SASA (Figure 3b).

7 Conclusion

This work introduces Generative Bayesian Optimization (GenBO), a unifying framework that turns any generative model into a sampler whose density tracks BO acquisition functions. We have shown that loss functions over generative models, such as DPO and KL divergences, can be applied to directly learn samplers for batch BO. By eliminating intermediate regression or classification surrogates, GenBO reduces approximation error, simplifies the pipeline to learning just a single generative model, and scales naturally to large batches and high-dimensional or combinatorial design spaces. Theoretical results show convergence to the target distribution, and experiments on text optimization and protein design tasks demonstrate competitive performance with more complex surrogate-guided baselines. A few challenges remain. For some variants, GenBO requires choosing and fixing the prior before optimization, and its performance depends on sensible settings of utility and temperature parameters, whose theory could be further explored. Another avenue is the adaptation to acquisition strategies not expressible as expected utilities, such as Thompson sampling and upper confidence bound. Despite these caveats, GenBO's minimal moving parts and principled acquisition-driven training mark a simpler and more scalable alternative to multi-stage guided generation methods.

References

Yasin Abbasi-Yadkori. *Online Learning for Linearly Parametrized Control Problems*. PhD, University of Alberta, 2012.

Dhruv Agarwal, Manoj Ghuhan Arivazhagan, Rajarshi Das, Sandesh Swamy, Sopan Khosla, and Rashmi Gangadharaiah. Searching for optimal solutions with LLMs via bayesian optimization. In *The Thirteenth International Conference on Learning Representations*, 2025.

Virginia Aglietti, Ira Ktena, Jessica Schrouff, Eleni Sgouritsa, Francisco Ruiz, Alan Malek, Alexis Bellot, and Silvia Chiappa. FunBO: Discovering acquisition functions for bayesian optimization with funsearch. In *Forty-second International Conference on Machine Learning*, 2025.

Javad Azimi, Alan Fern, and Xiaoli Z. Fern. Batch bayesian optimization via simulation matching. In *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010 (NIPS 2010)*, 2010.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback. *arXiv e-prints*, art. arXiv:2204.05862, April 2022. doi: 10.48550/arXiv.2204.05862.

Heejong Bong and Alessandro Rinaldo. Generalized results for the existence and consistency of the MLE in the Bradley-Terry-Luce model. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 2160–2177. PMLR, 2022.

Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration inequalities: A Nonasymptotic Theory of Independence*. Oxford University Press, 2013.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

David Brookes, Hahnbeom Park, and Jennifer Listgarten. Conditioning by adaptive sampling for robust design. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 773–782, Long Beach, CA, USA, 2019. PMLR.

Roberto Calandra, André Seyfarth, Jan Peters, and Marc Peter Deisenroth. Bayesian optimization for learning gaits under uncertainty. *Annals of Mathematics and Artificial Intelligence*, 76 (1):5–23, 2016. doi: 10.1007/s10472-015-9463-9. URL https://doi.org/10.1007/s10472-015-9463-9.

Angelica Chen, Samuel Don Stanton, Frances Ding, Robert G Alberstein, Andrew Martin Watkins, Richard Bonneau, Vladimir Gligorijevic, Kyunghyun Cho, and Nathan C. Frey. Generalists vs. specialists: Evaluating LLMs on highly-constrained biophysical sequence optimization tasks. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 9029–9072. PMLR, 13–19 Jul 2025.

Sayak Ray Chowdhury and Aditya Gopalan. On Kernelized Multi-armed Bandits. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, Sydney, Australia, 2017.

Sayak Ray Chowdhury, Anush Kini, and Nagarajan Natarajan. Provably robust DPO: Aligning language models with noisy feedback. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 42258–42274. PMLR, 2024.

Jaewon Chu, Jinyoung Park, Seunghun Lee, and Hyunwoo J. Kim. Inversion-based latent bayesian optimization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=TrN5TcWY87.

Zhongxiang Dai, Yao Shu, Bryan Kian Hsiang Low, and Patrick Jaillet. Sample-then-optimize batch neural Thompson sampling. In *36th Conference on Neural Information Processing Systems (NeurIPS 2022)*, New Orleans, LA, USA, 2022.

Bernard Delyon and François Portier. Asymptotic optimality of adaptive importance sampling. In S Bengio, H Wallach, H Larochelle, K Grauman, N Cesa-Bianchi, and R Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31, Montréal, Canada, 2018. Curran Associates, Inc.

Audrey Durand, Odalric-Ambrym Maillard, and Joelle Pineau. Streaming kernel regression with provably adaptive mean, variance, and regularization. *Journal of Machine Learning Research*, 19 (1):650–683, 2018.

Peter I. Frazier and Jialei Wang. *Bayesian Optimization for Materials Design*, pp. 45–75. Springer International Publishing, Cham, 2016. ISBN 978-3-319-23871-5. doi: 10.1007/978-3-319-23871-5_3. URL https://doi.org/10.1007/978-3-319-23871-5_3.

Roman Garnett. *Bayesian Optimization*. Cambridge University Press, 2023. URL https://bayesoptbook.com/.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

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. doi: 10.1021/acscentsci.7b00572.

Miguel González-Duque, Richard Michael, Simon Bartels, Yevgen Zainchkovskyy, Søren Hauberg, and Wouter Boomsma. A survey and benchmark of high-dimensional Bayesian optimization of discrete sequences. In *The 38th Conference on Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, Vancouver, Canada, 2024.

Miguel González-Duque, Simon Bartels, and Richard Michael. Poli: a libary of discrete sequence objectives, 2024. URL https://github.com/MachineLearningLifeScience/poli.

Nate Gruver, Samuel Stanton, Nathan Frey, Tim G.J. Rudner, Isidro Hotzel, Julien Lafrance-Vanasse, Arvind Rajpal, Kyunghyun Cho, and Andrew Gordon Wilson. Protein design with guided discrete diffusion. In *Advances in Neural Information Processing Systems*, volume 36, New Orleans, LA, USA, 2023.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034, 2015.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: convergence and generalization in neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pp. 8580–8589, Red Hook, NY, USA, 2018. Curran Associates Inc.

Kirthevasan Kandasamy, Akshay Krishnamurthy, Jeff Schneider, and Barnabas Poczos. Asynchronous parallel Bayesian optimisation via Thompson sampling. In *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics (AISTATS)*, Lanzarote, Spain, 2018.

Siddarth Krishnamoorthy, Satvik Mehul Mashkaria, and Aditya Grover. Diffusion models for black-box optimization. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 17842–17857. PMLR, 2023.

Seunghun Lee, Jinyoung Park, Jaewon Chu, Minseo Yoon, and Hyunwoo J. Kim. Latent bayesian optimization via autoregressive normalizing flows. In *The Thirteenth International Conference on Learning Representations*, 2025.

Yucen Lily Li, Tim G. J. Rudner, and Andrew Gordon Wilson. A study of Bayesian neural network surrogates for Bayesian optimization. In *2024 International Conference on Learning Representations (ICLR)*, Vienna, Austria, 2024. OpenReview.

Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky T. Q. Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code. *arXiv e-prints*, 2024.

Tennison Liu, Nicolás Astorga, Nabeel Seedat, and Mihaela van der Schaar. Large language models to enhance bayesian optimization. In *The Twelfth International Conference on Learning Representations*, 2024.

Henry Moss, Sebastian W. Ober, and Tom Diethe. Return of the latent space COWBOYS: Rethinking the use of VAEs for bayesian optimisation of structured spaces. In *Forty-second International Conference on Machine Learning*, 2025. URL `https://openreview.net/forum?id=U354tbTjav`.

Rafael Oliveira, Lionel Ott, and Fabio Ramos. No-regret approximate inference via Bayesian optimisation. In *37th Conference on Uncertainty in Artificial Intelligence (UAI 2021)*, 2021.

Rafael Oliveira, Louis Tiao, and Fabio T. Ramos. Batch bayesian optimisation via density-ratio estimation with guarantees. *Advances in Neural Information Processing Systems*, 35:29816–29829, 2022.

Mary Phuong and Marcus Hutter. Formal algorithms for transformers. *arXiv preprint arXiv:2207.09238*, 2022.

Gilles Pisier. Subgaussian sequences in probability and Fourier analysis. *Graduate Journal of Mathematics*, 1:59–78, 2016.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in neural information processing systems*, volume 36, pp. 53728–53741, 2023.

Carl E. Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, Cambridge, MA, 2006.

Saburou Saitoh and Yoshihiro Sawano. *Theory of Reproducing Kernels and Applications*. Springer, 2016.

Joost Schymkowitz, Jesper Borg, Francois Stricher, Robby Nys, Frederic Rousseau, and Luis Serrano. The foldx web server: an online force field. *Nucleic Acids Research*, 33:W382–W388, July 2005. ISSN 0305-1048. doi: 10.1093/nar/gki387.

Nihar B. Shah, Sivaraman Balakrishnan, Joseph Bradley, Abhay Parekh, Kannan Ramchandran, and Martin J. Wainwright. Estimation from pairwise comparisons: Sharp minimax bounds with topology dependence. *Journal of Machine Learning Research*, 17, 2016.

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando De Freitas. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1): 148–175, 2016.

Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical bayesian optimization of machine learning algorithms. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 25*, pp. 2951–2959. Curran Associates, Inc., 2012.

Jiaming Song, Lantao Yu, Willie Neiswanger, and Stefano Ermon. A general recipe for likelihood-free Bayesian optimization. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, Baltimore, Maryland, USA, 2022. PMLR 162.

Samuel Stanton, Wesley Maddox, Nate Gruver, Phillip Maffettone, Emily Delaney, Peyton Greenside, and Andrew Gordon Wilson. Accelerating Bayesian optimization for biological sequence design with denoising autoencoders. In *International Conference on Machine Learning*, pp. 20459–20478. PMLR, 2022.

Samuel Stanton, Robert Alberstein, Nathan Frey, Andrew Watkins, and Kyunghyun Cho. Closed-form test functions for biophysical sequence optimization algorithms. *arXiv preprint arXiv:2407.00236*, 2024.

Daniel M. Steinberg, Rafael Oliveira, Cheng Soon Ong, and Edwin V. Bonilla. Variational search distributions. In *The Thirteenth International Conference on Learning Representations*, Singapore, 2025.

Ingo Steinwart and Andreas Christmann. Kernels and reproducing kernel Hilbert spaces. In *Support Vector Machines*, chapter 4, pp. 110–163. Springer, New York, NY, 2008.

Miroslav Štrupl, Francesco Faccio, Dylan R Ashley, Rupesh Kumar Srivastava, and Jürgen Schmidhuber. Reward-weighted regression converges to a global optimum. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 8361–8369, 2022.

Shion Takeno, Hitoshi Fukuoka, Yuhki Tsukada, Toshiyuki Koyama, Motoki Shiga, Ichiro Takeuchi, and Masayuki Karasuyama. Multi-fidelity Bayesian optimization with max-value entropy search and its parallelization. In *37th International Conference on Machine Learning (ICML 2020)*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9276–9287. PMLR, 2020.

Felix Teufel, Carsten Stahlhut, and Jesper Ferkinghoff-Borg. Batched energy-entropy acquisition for Bayesian optimization. In *38th Conference on Neural Information Processing Systems (NeurIPS 2024)*, Vancouver, Canada, 2024.

Louis C. Tiao, Aaron Klein, Matthias Seeger, Edwin V. Bonilla, Cedric Archambeau, and Fabio Ramos. BORE: Bayesian optimization by density-ratio estimation. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*. PMLR, 2021.

Austin Tripp, Erik Daxberger, and José Miguel Hernández-Lobato. Sample-efficient optimization in the latent space of deep generative models via weighted retraining. *Advances in Neural Information Processing Systems*, 33:11259–11272, 2020.

James T. Wilson, Frank Hutter, and Marc Peter Deisenroth. Maximizing acquisition functions for Bayesian optimization. In *32nd Conference on Neural Information Processing Systems (NeurIPS 2018)*, Montréal, Canada, 2018.

Taeyoung Yun, Kiyoung Om, Jaewoo Lee, Sujin Yun, and Jinkyoo Park. Posterior Inference with Diffusion Models for High-dimensional Black-box Optimization. In *Forty-second International Conference on Machine Learning (ICML)*, Vancouver, Canada, 2025. URL https://openreview.net/forum?id=EXds2NBOoq.

A Gaussian processes for BO

Assume a Gaussian process prior over f, e.g., $f \sim \mathcal{GP}(0, k)$, where $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a positive-definite kernel (Rasmussen & Williams, 2006). Then, given a set of observations $\mathcal{D}_n := \{\mathbf{x}_i, y_i\}_{i=1}^n$, corrupted by Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$, the posterior $f|\mathcal{D}_n \sim \mathcal{GP}(\hat{f}_n, k_n)$ is available in closed form with mean and covariance function given by:

$$\hat{f}_n(\mathbf{x}) := \mathbf{k}_n(\mathbf{x})^\mathsf{T}(\mathbf{K}_n + \sigma_\epsilon^2\mathbf{I})^{-1}\mathbf{y}_n \tag{26}$$

$$k_n(\mathbf{x}, \mathbf{x}') := k(\mathbf{x}, \mathbf{x}') - \mathbf{k}_n(\mathbf{x})^\mathsf{T}(\mathbf{K}_n + \sigma_\epsilon^2\mathbf{I})^{-1}\mathbf{k}_n(\mathbf{x}') \tag{27}$$

$$\sigma_n^2(\mathbf{x}) := k_n(\mathbf{x}, \mathbf{x}), \tag{28}$$

where $\mathbf{k}_n(\mathbf{x}) := [k(\mathbf{x}, \mathbf{x}_i)]_{i=1}^n \in \mathbb{R}^n$, $\mathbf{K}_n := [k(\mathbf{x}_i, \mathbf{x}_j)]_{i,j=1}^n \in \mathbb{R}^{n\times n}$, $\mathbf{y}_n := [y_i]_{i=1}^n \in \mathbb{R}^n$, for $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$. With these closed-form expressions, GP models allow BO algorithms to quantify uncertainty and assess expected utilities of their decisions. However, note that, due to matrix inversions, exact GP inference incurs a computational cost of $\mathcal{O}(n^3)$. Hence, one often has to resort to low-rank approximations to make GP predictions tractable in cases involving large amounts of data, such as batch evaluations with large batch size. Alternatively, one may completely discard the GP models and use other surrogates, such as neural networks, and there has been an increasing literature on how to reliably quantify uncertainty for BO when using these models (Li et al., 2024).

B The GenBO algorithm

Algorithm 1: GenBO

```
Input: Domain $\mathcal{X}$, initial data $\mathcal{D}_0$
for $t \in \{1, \dots, T\}$ do
    $q_t \in \mathrm{argmin}_{q\in\mathcal{Q}} L_{t-1}(q)$                      // Fit proposal distribution
    $\mathcal{B}_t \overset{i.i.d.}{\sim} q_t$                                   // Sample batch
    $y_{t,i} \leftarrow f(\mathbf{x}_{t,i}) + \epsilon_{t,i}$, for $i \in \{1, \dots, B\}$   // Collect observations
    $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{\mathbf{x}_{t,i}, y_{t,i}\}_{i=1}^B$               // Update data
```

C Learning parametric models with RKHS convex losses

In this section, we consider the general problem of learning a function g_* with a parametric model $g : \mathcal{X} \times \Theta \to \mathbb{R}$, where the parameter space Θ is an arbitrary finite-dimensional vector space. Most existing results in the Bayesian optimization and bandits literature for learning these models from inherently dependent data are only valid for linear models or kernel machines. As we will consider arbitrary generative models, we need to derive convergence results applicable to a wider class models, accommodating popular modern frameworks. To do so, we will not assume identifiability, so that it is not necessary that some $\theta_* \in \Theta$ exists such that $g_* = g(\cdot; \theta_*)$. Instead, we will replace identifiability with a much milder assumption that g_* lies in a reproducing kernel Hilbert space (RKHS) large enough to also contain the models, as described next.

Assumption A1. *The true function $g_* : \mathcal{X} \to \mathbb{R}$ is a member of a reproducing kernel Hilbert space $\mathcal{H}_k$, associated with a positive-semidefinite kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, which is bounded $\sup_{\mathbf{x}\in\mathcal{X}} k(\mathbf{x}, \mathbf{x}) \leq b_k^2$ for a given $b_k > 0$. In addition, we assume that the models can also be found as elements of the same RKHS, i.e., $\{g(\cdot; \theta) \mid \theta \in \Theta\} \subset \mathcal{H}_k$.*

The assumption above allows us to consider functions g_* which cannot be perfectly approximated by the model, though which yet live in the same underlying Hilbert space $\mathcal{H}_k$. The reproducing kernel assumption is also mild, as it simply means that function evaluations are continuous (i.e., well behaved), which can usually not be guaranteed in other types of Hilbert spaces, such as, e.g., $\mathcal{L}_2$-spaces. In fact, every Hilbert space of functions where evaluation functionals are continuous is an RKHS by definition (Steinwart & Christmann, 2008, Def. 4.18). Lastly, we note that we can always find a RKHS that contains the models, such as the minimal construction below.

Lemma 2. *Let $g : \mathcal{X} \times \Theta \to \mathbb{R}$ represent a class of models parameterized by $\theta \in \Theta$. Assume that $g(\mathbf{x}; \cdot) \in \mathcal{H}_\Theta$, for all $\mathbf{x} \in \mathcal{X}$, where $\mathcal{H}_\Theta$ is a reproducing kernel Hilbert space associated with a positive-definite kernel $k_\Theta : \Theta \times \Theta \to \mathbb{R}$. It then follows that:*

$$\mathcal{H}_g := \{h : \mathcal{X} \to \mathbb{R} \mid \exists w \in \mathcal{H}_\Theta : h(\mathbf{x}) = \langle w, g(\mathbf{x}, \cdot)\rangle_{\mathcal{H}_\Theta}, \forall \mathbf{x} \in \mathcal{X}\} \tag{29}$$

equipped with the norm:

$$\|h\|_{\mathcal{H}_g} := \inf\{\|w\|_{\mathcal{H}_\Theta} : w \in \mathcal{H}_\Theta, h(\mathbf{x}) = \langle w, g(\mathbf{x}, \cdot)\rangle_{\mathcal{H}_\Theta}, \forall \mathbf{x} \in \mathcal{X}\} \tag{30}$$

constitutes the unique RKHS for which $k_g : (\mathbf{x}, \mathbf{x}') \mapsto \langle g(\mathbf{x}, \cdot), g(\mathbf{x}', \cdot)\rangle_{\mathcal{H}_\Theta}$ is the reproducing kernel.

Proof. This is a direct application of classic RKHS results (e.g., Steinwart & Christmann, 2008, Thm. 4.21) where we are treating $\phi : \mathbf{x} \mapsto g(\mathbf{x}, \cdot)$ as a feature map mapping into an existing Hilbert space $\mathcal{H}_\Theta$ and taking advantage of its structure to define a new one. □

Remark 2. The RKHS $\mathcal{H}_g$ described above has the special property that for any $\theta \in \Theta$, the RKHS norm of the model is given by:

$$\|g(\cdot, \theta)\|^2_{\mathcal{H}_g} = k_\Theta(\theta, \theta)\,, \tag{31}$$

since $\langle k_\Theta(\cdot, \theta), g(\mathbf{x}, \cdot)\rangle_{\mathcal{H}_\Theta} = g(\mathbf{x}, \theta)$ for all $\mathbf{x} \in \mathcal{X}$, and $k_\Theta(\cdot, \theta)$ is the unique representation of the evaluation functional at θ in the RKHS $\mathcal{H}_\Theta$. The rest follows from the definition in Equation 30. Hence, each choice of k_Θ gives us a potential RKHS norm regularizer.

Remark 3. If the RKHS in Lemma 2 is insufficiently small to contain the function g_* of interest, we can always combine two RKHS to produce a third one containing all elements of the two. Namely, if $g_* \in \mathcal{H}_* \neq \mathcal{H}_g$ with kernel $k_* : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, we can define $k := k_* + k_g$, so that $\mathcal{H}_k := \mathcal{H}_* \oplus \mathcal{H}_g$ is also a RKHS (Steinwart & Christmann, 2008; Saitoh & Sawano, 2016).

Definition 1 (Strong convexity). *A differentiable function $f : \mathcal{H} \to \mathbb{R}$ on a Hilbert space $\mathcal{H}$ is α-strongly convex, for $\alpha > 0$, if:*

$$\forall h, h', \quad f(h) \geq f(h') + \langle \nabla f(h'), h - h'\rangle + \frac{\alpha}{2}\|h - h'\|^2_{\mathcal{H}}\,.$$

Definition 2 (Smoothness). *A function $f : \mathcal{H} \to \mathcal{Y}$ between Hilbert spaces $\mathcal{H}$ and $\mathcal{Y}$ is η-smooth if:*

$$\forall h, h', \quad \|f(h) - f(h')\|_{\mathcal{Y}} \leq \eta\|h - h'\|_{\mathcal{H}}\,. \tag{32}$$

Definition 3 (Sub-Gaussianity). *A real-valued random variable ϵ is said to be σ_ϵ-sub-Gaussian if:*

$$\forall s \in \mathbb{R}, \quad \mathbb{E}[\exp(s\epsilon)] \leq \exp\left(\frac{s^2\sigma^2_\epsilon}{2}\right). \tag{33}$$

In addition, a real-valued stochastic process $\{\epsilon_t\}_{t=1}^\infty$ adapted to a filtration $\{\mathfrak{F}_t\}_{t=0}^\infty$ is conditionally σ^2_ϵ-sub-Gaussian if the following almost surely holds:

$$\forall s \in \mathbb{R}, \quad \mathbb{E}[\exp(s\epsilon_t) \mid \mathfrak{F}_{t-1}] \leq \exp\left(\frac{s^2\sigma^2_\epsilon}{2}\right), \quad \forall t \in \mathbb{N}\,. \tag{34}$$

Assumption A2 (Regularization). *The regularizer $\bar{R}_n : \mathcal{H}_k \to \mathbb{R}$ is λ-strongly convex, twice differentiable, and has η_0-smooth gradients.*

Common choices of regularization scheme, such as the squared norm $\|g\|^2_k$, suffice the assumptions above. Strong convexity does not require a function to be twice differentiable, but such assumption can greatly simplify the analysis and it is common in modern deep learning frameworks.

Assumption A3 (Loss). *For any $y \in \mathbb{R}$, the point loss $\ell_y := \ell(\cdot, y) : \mathbb{R} \to \mathbb{R}$ is α_ℓ-strongly convex, twice differentiable, and has η_ℓ-smooth first-order derivatives. In addition, given any $m \in \mathcal{H}_k$, we assume the first-order derivative $\dot{\ell}_y(m(g_*))$ is conditionally σ_ℓ-sub-Gaussian when $y \sim p(y|m(g_*))$.*

Note that most loss functions in the deep learning literature satisfy the assumptions above, including the squared error and the cross entropy loss. The original Bradley-Terry model in the DPO paper (Rafailov et al., 2023) is not strongly convex, whereas its robust version (Chowdhury et al., 2024), which accounts for preference noise, can be shown to satisfy strong convexity and smoothness.

Now consider we have access to observation data of the form $\mathcal{D}_n := \{m_i, y_i\}_{i=1}^n$, where $y_i \sim p(y|m_i(g_*))$, and $m_i : \mathcal{H}_k \to \mathbb{R}$ represents a bounded linear observation functional, e.g., $m_i(g_*) = m_i(g_*)$, or $m_i(g_*) = g_*(\mathbf{x}_{i,1}) - g_*(\mathbf{x}_{i,2})$, etc., for $i \in \{1, \dots, n\}$, which follow the true (unknown) function $g_* : \mathcal{X} \to \mathbb{R}$. The data are not i.i.d., and each m_t is the result of an algorithmic decision based on the currently available dataset $\mathcal{D}_{t-1}$ and a model $g(\cdot, \theta_t)$ learned from it. The model is learned by minimizing a loss function:

$$L_n(g_\theta) := \bar{R}_n(g_\theta) + \sum_{i=1}^n \ell(m_i(g_*), y_i)\,, \tag{35}$$

where $\bar{R}_n : \mathcal{H}_k \to \mathbb{R}_+$ is a regularization term, and $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ encodes the data dependency. Despite the definition of a regularization term over $\mathcal{H}_k$, following Remark 2, we can use any positive-definite kernel $k_\Theta : \Theta \times \Theta \to \mathbb{R}$ compatible with Lemma 2 to set $\bar{R}_n$ such that:

$$\bar{R}_n(g_\theta) = \lambda \|g_\theta\|^2_{\mathcal{H}_g} = \lambda k_\Theta(\theta, \theta)\,. \tag{36}$$

In this case, a quadratic regularization term $\|\theta\|_2^2$ corresponds to the choice of a linear kernel, i.e., $k_\Theta(\theta, \theta') = \theta \cdot \theta'$, which might appear quite restrictive, as it assumes that our models are linear functions of the parameters. However, note that, for overparameterized neural networks, at the infinite-width limit the model is actually linear in the parameters (Jacot et al., 2018). If we want to be more parsimonious, alternatively, we can choose k_Θ as a universal kernel, such as the squared-exponential, yet preferably not translation invariant, so that $k_\Theta(\theta, \theta)$ is not a constant. One kernel satisfying such assumption would be the exponential dot-product kernel $k_\Theta(\theta, \theta') := \exp(\theta \cdot \theta')$, which is universal for continuous functions over compact subsets of Θ. Nevertheless, we do not impose restrictions on the form of the regularization term $\bar{R}_n$, except for the one below, which is followed by our assumptions on the loss.

We can now analyze the approximation error with respect to g_* for the following estimators:[1]

$$\theta_n \in \operatorname*{argmin}_{\theta \in \Theta} L_n(g_\theta) \tag{37}$$

$$g_n \in \operatorname*{argmin}_{g \in \mathcal{H}_k} L_n(g)\,. \tag{38}$$

The first one gives us the best parametric approximation g_{θ_n} based on the data and is what our algorithm will use. The second estimator corresponds to the non-parametric approximation, which we will use as a tool for our analysis, and not assume as a component of the algorithm. The assumptions above allow us to bound distances between these estimators and the true g_* as a function of the loss and gradient values.

Lemma 1. *Let Assumption A1, A2 and A3 be satisfied. Then, for any $g \in \mathcal{H}_k$, the following holds:*

$$\frac{1}{2}\|g - g_n\|^2_{H_n} \leq L_n(g) - L_n(g_n) \leq \frac{1}{2}\|\nabla L_n(g)\|^2_{H_n^{-1}}\,, \tag{24}$$

where $H_n : \mathcal{H}_k \to \mathcal{H}_k$ is an operator-valued lower bound on the Hessian of the loss L_n:

$$\forall g \in \mathcal{H}_k, \quad \nabla^2 L_n(g) \succeq H_n := \lambda I + \alpha_\ell \sum_{i=1}^n m_i \otimes m_i\,, \tag{25}$$

where $\phi(\mathbf{x}) := k(\cdot, \mathbf{x})$, for $\mathbf{x} \in \mathcal{X}$.

Proof of Lemma 1. We note that the Hessian of the losses can be lower bounded by:

$$\begin{aligned}\forall g \in \mathcal{H}_k, \qquad \nabla_g^2 \ell(m(g), y) &= \ddot{\ell}_y(m(g)) \nabla_g m(g) \otimes \nabla_g m(g) + \dot{\ell}_y(m(g)) \nabla_g^2 m(g) \\ &= \ddot{\ell}_y(m(g)) m \otimes m \\ &\succeq \alpha_\ell m \otimes m, \qquad \forall y \in \mathbb{R}, \forall \mathbf{x} \in \mathcal{X}\,,\end{aligned} \tag{39}$$

where we applied the fact that $\nabla_g m(g) = \nabla_g \langle g, m \rangle_k = m$, and the second derivatives $\ddot{\ell}_y$ of the loss function $\ell(\cdot, y)$ have a positive lower bound due to the strong convexity assumption (A3). Combining with Assumption A2, we get:

$$\forall g \in \mathcal{H}_k, \qquad \nabla_g^2 L_n(g) \succeq \lambda I + \alpha_\ell \sum_{i=1}^n m_i \otimes m_i =: H_n\,. \tag{40}$$

[1]We are implicitly assuming that such global optima exist. This is true for the optimization in $\mathcal{H}_k$, as we will show, but not always guaranteed for the optimization over Θ.

Now applying a first order Taylor expansion to L_n at any $g \in \mathcal{H}_k$, the error term is controlled by the Hessian ∇_g^2 at an intermediate point $\bar{g} \in \mathcal{H}_k$, which is uniformly lower bounded by H_n. Expanding L_n around g_n, we then have that:

$$\begin{aligned}\forall g \in \mathcal{H}_k, \quad L_n(g) - L_n(g_n) &= \langle \nabla L_n(g_n), g - g_n \rangle + \frac{1}{2}\|g - g_n\|^2_{\nabla^2 L_n(\bar{g}_n)} \\ &\geq \frac{1}{2}\|g - g_n\|^2_{H_n}\,,\end{aligned} \tag{41}$$

where $\bar{g}_n = s g_n + (1-s)g$, for some $s \in [0,1]$, and we applied the Hessian inequality (40) and the fact that $\nabla L_n(g_n) = 0$, as g_n is a minimizer. Hence, the lower bound (24) follows. Conversely, by expanding L_n around any g and evaluating at g_n, we have:

$$\forall g \in \mathcal{H}_k, \quad L_n(g_n) = L_n(g) + \langle \nabla L_n(g), g_n - g \rangle + \frac{1}{2}\|g_n - g\|^2_{\nabla^2 L_n(\bar{g}'_n)} \tag{42}$$

Rearranging the terms yields:

$$\begin{aligned}\forall g \in \mathcal{H}_k, \quad L_n(g) - L_n(g_n) &= \langle \nabla L_n(g), g - g_n \rangle - \frac{1}{2}\|g - g_n\|^2_{\nabla^2 L_n(\bar{g}'_n)} \\ &\leq \sup_{\tilde{g} \in \mathcal{H}_k} \langle \nabla L_n(g), \tilde{g} \rangle - \frac{1}{2}\|\tilde{g}\|^2_{\nabla^2 L_n(\bar{g}'_n)} \\ &\leq \sup_{\tilde{g} \in \mathcal{H}_k} \langle \nabla L_n(g), \tilde{g} \rangle - \frac{1}{2}\|\tilde{g}\|^2_{H_n}\,,\end{aligned} \tag{43}$$

whose right-hand side is strongly concave and has a unique maximizer at:

$$\tilde{g} = H_n^{-1} \nabla L_n(g)\,. \tag{44}$$

Replacing this result into the previous equation finally leads us to the upper bound in Lemma 1. □

Lemma 1 allows us to control the approximation error by means of the functional gradients of L_n, without the need to know an explicit form for the optimal solution g_n. We can now proceed to derive our error bound, which will make use of the following result from the online learning literature.

Lemma 3 (Abbasi-Yadkori, 2012, Cor. 3.6)**.** *Let $\{\mathfrak{F}_t\}_{t=0}^\infty$ be an increasing filtration, $\{\epsilon_t\}_{t=1}^\infty$ be a real-valued stochastic process, and $\{\phi_t\}_{t=1}^\infty$ be a stochastic process taking values in a separable real Hilbert space $\mathcal{H}$, with both processes adapted to the filtration. Assume that $\{\phi_t\}_{t=1}^\infty$ is predictable w.r.t. the filtration, i.e., ϕ_t is $\mathfrak{F}_{t-1}$-measurable, and that ϵ_t is conditionally σ_ϵ^2-sub-Gaussian, for all $t \in \mathbb{N}$. Then, given any $\delta \in (0,1)$, with probability at least $1-\delta$,*

$$\forall t \in \mathbb{N}, \quad \left\|\sum_{i=1}^t \epsilon_i \phi_i\right\|^2_{(V + \Phi_t \Phi_t^\mathsf{T})^{-1}} \leq 2\sigma_\epsilon^2 \log\left(\frac{\det(\mathbf{I} + \Phi_t^\mathsf{T} V^{-1} \Phi_t)^{\frac{1}{2}}}{\delta}\right),$$

for any positive-definite operator $V \succ 0$ on $\mathcal{H}$, and where we set $\Phi_t := [\phi_1, \dots, \phi_t]$.

Theorem 1. *Consider the setting in Lemma 1, and let $\xi_k := \inf_{\theta \in \Theta}\|g(\cdot,\theta) - g_*\|_k$. Then, given any $\delta \in (0,1)$, the following holds with probability at least $1-\delta$:*

$$\forall n \in \mathbb{N}, \quad |\langle m, g_* \rangle_k - \langle m, g_{\theta_n} \rangle_k| \leq \|m\|_{H_n^{-1}}\left(2\beta_n(\delta) + \eta_0 \xi_k + b_k \eta_\ell \xi_k \sum_{i=1}^n \|m_i\|_{H_n^{-1}}\right), \forall m \in \mathcal{H}_k,$$

where $\beta_n(\delta) := \lambda^{-1/2}\|\nabla \bar{R}_n(g_*)\|_k + \sigma_\ell \sqrt{2\alpha_\ell^{-1} \log(\det(\mathbf{I} + \alpha_\ell \lambda^{-1} M_n^\mathsf{T} M_n)^{1/2}/\delta)}$, and $M_n := [m_1, \dots, m_n]$.

Proof of Theorem 1. Fix any $m \in \mathcal{H}_k$ and $n \in \mathbb{N}$, the approximation error can then be expanded as:

$$|\langle m, g_* \rangle_k - \langle m, g_{\theta_n} \rangle_k| \leq |\langle m, g_* \rangle_k - \langle m, g_n \rangle_k| + |\langle m, g_{\theta_n} \rangle_k - \langle m, g_n \rangle_k|\,. \tag{45}$$

By Lemma 1, for any $g \in \mathcal{H}_k$, we have that:

$$\begin{aligned}
|\langle m, g_n \rangle_k - \langle m, g \rangle_k| &= |\langle m, g_n - g \rangle_k| \\
&= |\langle H_n^{-1/2} m, H_n^{1/2}(g_n - g) \rangle_k| \\
&\leq \|H_n^{-1/2} m\| \|H_n^{1/2}(g_n - g)\| \\
&= \|m\|_{H_n^{-1}} \|g_n - g\|_{H_n} \\
&\leq \|m\|_{H_n^{-1}} \|\nabla L_n(g)\|_{H_n^{-1}} \,,
\end{aligned} \tag{46}$$

where the first inequality follows by Cauchy-Schwarz, and the last is due to Lemma 1. Expanding the gradient term, we have:

$$\begin{aligned}
\|\nabla L_n(g)\|_{H_n^{-1}} &= \left\| \nabla \bar{R}_n(g) + \sum_{i=1}^n \dot{\ell}_{y_i}(\langle m_i, g \rangle_k) m_i \right\|_{H_n^{-1}} \\
&\leq \|\nabla \bar{R}_n(g)\|_{H_n^{-1}} + \left\| \sum_{i=1}^n \dot{\ell}_{y_i}(\langle m_i, g \rangle_k) m_i \right\|_{H_n^{-1}} \\
&\leq \frac{1}{\sqrt{\lambda}} \|\nabla \bar{R}_n(g)\|_k + \left\| \sum_{i=1}^n \dot{\ell}_{y_i}(\langle m_i, g \rangle_k) m_i \right\|_{H_n^{-1}} \,,
\end{aligned} \tag{47}$$

where we applied the triangle inequality to obtain the second line and the fact that $H_n \succ \lambda I$ implies $H_n^{-1} \prec \lambda^{-1} I$ led to the last line. For $g := g_*$, we can then apply Lemma 3 to the noisy sum above by setting $\mathfrak{F}_t$ as the σ-algebra generated by the random variables $\{m_i, y_i\}_{i=1}^t$ and m_{t+1}, $\epsilon_t := \alpha_\ell^{-1/2} \dot{\ell}_{y_t}(\langle g_*, m_t \rangle_k)$ and $\phi_t := \alpha_\ell^{1/2} m_t$, for all $t \in \mathbb{N}$, which leads us to:

$$\left\| \sum_{i=1}^n \dot{\ell}_{y_i}(\langle m_i, g_* \rangle_k) m_i \right\|_{H_n^{-1}}^2 \leq \frac{2\sigma_\ell^2}{\alpha_\ell} \log \left(\frac{\det(\mathbf{I} + \alpha_\ell \lambda^{-1} M_n^\mathsf{T} M_n)^{\frac{1}{2}}}{\delta} \right) \tag{48}$$

which holds uniformly over all $n \in \mathbb{N}$ with probability at least $1 - \delta$. Hence, it follows that:

$$\forall n \in \mathbb{N}, \quad \|\nabla L_n(g_*)\|_{H_n^{-1}} \leq \frac{1}{\sqrt{\lambda}} \|\nabla \bar{R}_n(g_*)\|_k + \left\| \sum_{i=1}^n \dot{\ell}_{y_i}(\langle m_i, g_* \rangle_k) m_i \right\|_{H_n^{-1}} \leq \beta_n(\delta), \tag{49}$$

with probability at least $1 - \delta$, where we set:

$$\beta_n(\delta) := \frac{1}{\sqrt{\lambda}} \|\nabla \bar{R}_n(g_*)\|_k + \sigma_\ell \sqrt{\frac{2}{\alpha_\ell} \log \left(\frac{\det(\mathbf{I} + \alpha_\ell \lambda^{-1} \mathbf{K}_n)^{\frac{1}{2}}}{\delta} \right)}. \tag{50}$$

Therefore, the pointwise approximation error of the RKHS-optimal estimator g_n is bounded as:

$$\forall n \in \mathbb{N}, \quad |\langle m, g_n \rangle_k - \langle m, g_* \rangle_k| \leq \beta_n(\delta) \|m\|_{H_n^{-1}}, \quad \forall \mathbf{x} \in \mathcal{X} \,, \tag{51}$$

with probability at least $1 - \delta$, with $\mathbf{K}_n := M_n^\mathsf{T} M_n = [k(\mathbf{x}_i, \mathbf{x}_j)]_{i,j=1}^n$.

For the remaining term, we have that:

$$\begin{aligned}
|\langle m, g_{\theta_n} \rangle_k - \langle m, g_n \rangle_k| &\leq \|m\|_{H_n^{-1}} \|g_{\theta_n} - g_n\|_{H_n} \\
&\leq \|m\|_{H_n^{-1}} \sqrt{2(L_n(g_{\theta_n}) - L_n(g_n))} \,,
\end{aligned} \tag{52}$$

which follows from Lemma 1. We can bound the loss difference via the gap term ξ_k if we can relate it to g_{θ_n}, though note that it is not guaranteed that the infimum is achieved by any particular $\theta \in \Theta$. From the definition of the infimum, however, it is a simple consequence that:

$$\forall \Delta > 0, \quad \exists \theta_\Delta \in \Theta : \qquad \|g_{\theta_\Delta} - g_*\|_k \leq \xi_k + \Delta \,. \tag{53}$$

Therefore, as θ_n minimizes L_n over all Θ, picking some $\Delta > 0$, we have that any θ_Δ satisfying the condition above leads us to:

$$\begin{aligned}
L_n(g_{\theta_n}) - L_n(g_n) &\leq L_n(g_{\theta_\Delta}) - L_n(g_n) \\
&\leq \frac{1}{2} \|\nabla L_n(g_{\theta_\Delta})\|_{H_n^{-1}}^2 \\
&\leq \frac{1}{2} \left(\|\nabla \bar{R}_n(g_{\theta_\Delta})\|_k + \left\| \sum_{i=1}^n \dot{\ell}_{y_i}(\langle m_i, g_{\theta_\Delta} \rangle_k) m_i \right\|_{H_n^{-1}} \right)^2 ,
\end{aligned} \tag{54}$$

where we applied Lemma 1 to derive the second line and Equation 47 for the third line. Now each term above can be bounded in terms of the approximation gap $\xi_k + \Delta$. Firstly, given the smoothness of the regularization gradients (Assumption A2), observe that:

$$\begin{aligned}\|\nabla \bar{R}_n(g_{\theta_\Delta})\|_k &= \|\nabla \bar{R}_n(g_*) + \nabla \bar{R}_n(g_{\theta_\Delta}) - \nabla \bar{R}_n(g_*)\|_k \\ &\leq \|\nabla \bar{R}_n(g_*)\|_k + \|\nabla \bar{R}_n(g_{\theta_\Delta}) - \nabla \bar{R}_n(g_*)\|_k \\ &\leq \|\nabla \bar{R}_n(g_*)\|_k + \eta_0 \|g_{\theta_\Delta} - g_*\|_k \\ &\leq \|\nabla \bar{R}_n(g_*)\|_k + \eta_0(\xi_k + \Delta)\,,\end{aligned} \tag{55}$$

where we applied the triangle inequality and the definition of smoothness (cf. Definition 2). Secondly, for the sum term, by smoothness of the loss derivatives (Assumption A3), we have that:

$$\begin{aligned}\forall i \in \{1, \dots, n\}, \quad \dot{\ell}_{y_i}(\langle m_i, g_{\theta_\Delta}\rangle_k) &= \dot{\ell}_{y_i}(\langle m_i, g_*\rangle_k) + \dot{\ell}_{y_i}(\langle m_i, g_{\theta_\Delta}\rangle_k) - \dot{\ell}_{y_i}(\langle m_i, g_*\rangle_k) \\ &\leq \dot{\ell}_{y_i}(\langle m_i, g_*\rangle_k) + \eta_\ell |\langle m_i, g_{\theta_\Delta}\rangle_k - \langle m_i, g_*\rangle_k| \\ &= \dot{\ell}_{y_i}(\langle m_i, g_*\rangle_k) + \eta_\ell |\langle m_i, g_{\theta_\Delta} - g_*\rangle_k| \\ &\leq \dot{\ell}_{y_i}(\langle m_i, g_*\rangle_k) + \eta_\ell \|m_i\|_k \|g_{\theta_\Delta} - g_*\|_k \\ &\leq \dot{\ell}_{y_i}(\langle m_i, g_*\rangle_k) + \eta_\ell b_k(\xi_k + \Delta),\end{aligned} \tag{56}$$

where the first inequality follows by smoothness, the second is due to Cauchy-Schwarz, and the last follows by the definition of θ_Δ and the boundedness of the kernel (cf. Assumption A1). Hence, the sum is bounded as:

$$\begin{aligned}\left\|\sum_{i=1}^n \dot{\ell}_{y_i}(\langle m_i, g_{\theta_\Delta}\rangle_k) m_i\right\|_{H_n^{-1}} &\leq \left\|\sum_{i=1}^n (\dot{\ell}_{y_i}(\langle m_i, g_*\rangle_k) + b_k\eta_\ell(\xi_k + \Delta)) m_i\right\|_{H_n^{-1}} \\ &\leq \left\|\sum_{i=1}^n (\dot{\ell}_{y_i}(\langle m_i, g_*\rangle_k) m_i\right\|_{H_n^{-1}} + b_k\eta_\ell(\xi_k + \Delta)\left\|\sum_{i=1}^n m_i\right\|_{H_n^{-1}} \\ &\leq \left\|\sum_{i=1}^n (\dot{\ell}_{y_i}(\langle m_i, g_*\rangle_k) m_i\right\|_{H_n^{-1}} + b_k\eta_\ell(\xi_k + \Delta)\sum_{i=1}^n \|m_i\|_{H_n^{-1}}\,.\end{aligned} \tag{57}$$

Substituting the upper bounds in equations 55 and 57 into Equation 54 and applying the concentration inequality in Equation 49 yields:

$$\begin{aligned}\|\nabla L_n(g_{\theta_\Delta})\|_{H_n^{-1}} \leq \frac{1}{\sqrt{\lambda}}\|\nabla \bar{R}_n(g_*)\|_k + \eta_0(\xi_k + \Delta) + \left\|\sum_{i=1}^n (\dot{\ell}_{y_i}(\langle m_i, g_*\rangle_k) m_i\right\|_{H_n^{-1}} \\ + b_k\eta_\ell(\xi_k + \Delta)\sum_{i=1}^n \|m_i\|_{H_n^{-1}} \\ \leq \beta_n(\delta) + \eta_0(\xi_k + \Delta) + b_k\eta_\ell(\xi_k + \Delta)\sum_{i=1}^n \|m_i\|_{H_n^{-1}}\,,\end{aligned} \tag{58}$$

which holds with the same probability as Equation 49. Lastly, as the gradient bound above is valid for any $\Delta > 0$, we can take the limit as $\Delta \to 0$ and substitute the result back into Equation 52 to get the model approximation error bound:

$$|\langle m, g_{\theta_n}\rangle_k - \langle m, g_n\rangle_k| \leq \|m\|_{H_n^{-1}} \left(\beta_n(\delta) + \eta_0\xi_k + b_k\eta_\ell\xi_k \sum_{i=1}^n \|m_i\|_{H_n^{-1}}\right), \quad \forall \mathbf{x} \in \mathcal{X}\,, \tag{59}$$

which also holds uniformly over all $n \in \mathbb{N}$ with probability at least $1 - \delta$. Combining Equation 59 and 51 leads to the final result, which concludes the proof. □

Despite the model being potentially non-linear and the loss not being required to be least-squares, Theorem 1 shows that we recover the same kind of RKHS-based error bound found in the kernelized bandits literature (Chowdhury & Gopalan, 2017; Durand et al., 2018; Oliveira et al., 2021), up to an approximation gap ξ_k w.r.t. the true function g_*. If identifiability holds, we have $\xi_k = 0$, and we recover the usual bounds (Durand et al., 2018).[2] Alternatively, in the case of neural networks, we can increase the width of the network over time (making sure the model scales up with the data is not uncommon in deep learning approaches), which would then lead to the model covering a whole RKHS, determined by the NTK (Jacot et al., 2018). In general, for a rich enough model class, one may expect ξ_k to be small.

Regarding the term resembling a pointwise predictive variance, by an application of Woodbury's identity, we have that:

$$\begin{aligned}
\|m\|^2_{H_n^{-1}} &= m^\mathsf{T}(\lambda I + \alpha_\ell M_n M_n^\mathsf{T})^{-1} m \\
&= m^\mathsf{T}(\lambda^{-1} I - \lambda^{-2} M_n (\alpha_\ell^{-1}\mathbf{I} + \lambda^{-1} M_n^\mathsf{T} M_n)^{-1} M_n^\mathsf{T}) m \\
&= \lambda^{-1} m^\mathsf{T}(I - M_n(\lambda\alpha_\ell^{-1}\mathbf{I} + M_n^\mathsf{T} M_n)^{-1} M_n^\mathsf{T}) m \\
&= \lambda^{-1}(\|m\|_k^2 - m^\mathsf{T} M_n (\lambda\alpha_\ell^{-1}\mathbf{I} + M_n^\mathsf{T} M_n)^{-1} M_n^\mathsf{T} m)\,,
\end{aligned} \tag{60}$$

If observations correspond to pointwise evaluations $m := k(\cdot, \mathbf{x})$ and $m_i := k(\cdot, \mathbf{x}_i)$, for $\mathbf{x} \in \mathcal{X}$ and $\{\mathbf{x}_i\}_{i=1}^n \subset \mathcal{X}$, we end up with:

$$\begin{aligned}
\|m\|^2_{H_n^{-1}} &= \|k(\cdot, \mathbf{x})\|^2_{H_n^{-1}} \\
&= \lambda^{-1}(k(\mathbf{x}, \mathbf{x}) - \mathbf{k}_n(\mathbf{x})^\mathsf{T}(\lambda\alpha_\ell^{-1}\mathbf{I} + \mathbf{K}_n)^{-1}\mathbf{k}_n(\mathbf{x})) \\
&= \lambda^{-1}\sigma_n^2(\mathbf{x})\,,
\end{aligned} \tag{61}$$

which corresponds to a scaled version of the posterior predictive variance $\sigma_n^2(\mathbf{x}) := k_n(\mathbf{x}, \mathbf{x})$ of a GP model (cf. Equation 27). We also have the following auxiliary result from VSD.

Lemma 4 (GP variance upper bound (Steinberg et al., 2025, Lem. E.5)). *Let $\{\mathbf{x}_n\}_{n\geq 1}$ be a sequence of $\mathcal{X}$-valued random variables adapted to the filtration $\{\mathfrak{F}_n\}_{n\geq 1}$. For a given $\mathbf{x} \in \mathcal{X}$, assume that the following holds:*

$$\exists T_* \in \mathbb{N} : \quad \forall T \geq T_*, \quad \sum_{n=1}^{T} \mathbb{P}\left[\mathbf{x}_n = \mathbf{x} \mid \mathfrak{F}_{n-1}\right] \geq b_T > 0\,, \tag{62}$$

for a some sequence of lower bounds $\{b_n\}_{n\in\mathbb{N}}$. Then, for a bounded kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ given observations at $\{\mathbf{x}_i\}_{i=1}^n$, the following holds with probability 1:

$$\sigma_n^2(\mathbf{x}) \in \mathcal{O}(b_n^{-1}). \tag{63}$$

In addition, if $b_n \to \infty$, then $\lim_{n\to\infty} b_n\sigma_n^2(\mathbf{x}) \leq \sigma_\epsilon^2$.

D Additional Experimental Detail

D.1 Text optimization

We use the same annealing threshold scheme for setting τ_t as Steinberg et al. (2025, Eqn. 20), where we set η such that when begin at $p_0 = 0.5$ we end at $p_T = 0.99$. For the proposal distribution, we found these short sequences we best generated by the simple mean-field categorical model,

$$q(\mathbf{x}|\phi) = \prod_{m=1}^{M} \mathrm{Categ}(x_m|\mathrm{softmax}(\phi_m)) \tag{64}$$

where $x_m \in \mathcal{V}$ and $\phi_m \in \mathbb{R}^{|\mathcal{V}|}$, and we directly optimize ϕ. VSD and CbAS use the simple MLP classifier guide in Figure 4.

[2] If we further assume that the model can represent *any* $g \in \mathcal{H}_k$, the factor of 2 multiplying β_n would also disappear, as the extra β_n arises from a bound over $\|g_{\theta_n} - g_n\|$, which would vanish.

```
Sequential(
    Embedding(
        num_embeddings=A,
        embedding_dim=16
    ),
    Dropout(p=0.1),
    Flatten(),
    LeakyReLU(),
    Linear(
        in_features=16 * M,
        out_features=64
    ),
    LeakyReLU(),
    Linear(
        in_features=64,
        out_features=1
    ),
)
```

(a) MLP architecture

```
Sequential(
    Embedding(
        num_embeddings=A,
        embedding_dim=E
    ),
    Dropout(p=0.2),
    Conv1d(
        in_channels=E,
        out_channels=C,
        kernel_size=Kc,
    ),
    LeakyReLU(),
    MaxPool1d(
        kernel_size=Kx,
        stride=Sx,
    ),
    Conv1d(
        in_channels=C,
        out_channels=C,
        kernel_size=Kc,
    ),
    LeakyReLU(),
    MaxPool1d(
        kernel_size=Kx,
        stride=Sx,
    ),
    Flatten(),
    LazyLinear(
        out_features=H
    ),
    LeakyReLU(),
    Linear(
        in_features=H,
        out_features=1
    ),
)
```

(b) CNN architecture

Figure 4: Classifier architectures used for VSD and CbAS in the experiments using PyTorch syntax. $\mathtt{A} = |\mathcal{V}|$, $\mathtt{M} = M$, and we give all other parameters in Table 2 if not directly indicated.

D.2 Protein design

We use the same threshold function and setting for all of the protein design experiments as in Section D.1. However, these tasks require a more sophisticated generative model that can capture local and global relationships that relate to protein's 3D structure. For this we use the auto-regressive (causal) transformer architecture also used in Steinberg et al. (2025),

$$q(\mathbf{x}|\phi) = \mathrm{Categ}(x_1|\mathrm{softmax}(\phi_1)) \prod_{m=2}^{M} q(x_m|x_{1:m-1}, \phi_{1:m}) \quad \text{where,}$$
$$q(x_m|x_{1:m-1}, \phi_{1:m}) = \mathrm{Categ}(x_m|\mathrm{DTransformer}(x_{1:m-1}, \phi_{1:m})). \tag{65}$$

See for the latter see Phuong & Hutter (2022, Algorithm 10 & Algorithm 14) for maximum likelihood training and sampling implementation details respectively. We give the architectural configuration for the transformers in each task in Table 1, and the classifier CNN used by VSD and CbAS is in Figure 4.

We use the following Ehrlich function configurations:

$M = 15$**:** motif length = 4, no. motifs = 2, quantization = 4

$M = 32$**:** motif length = 4, no. motifs = 2, quantization = 4

Configuration	Stability	SASA	Ehrlich 15	Ehrlich 32	Ehrlich 64
Layers	2	2	2	2	2
Feedforward network	256	256	32	64	128
Attention heads	4	4	1	2	3
Embedding size	64	64	10	20	30

Table 1: Transformer backbone configuration.

Configuration	Stability	SASA	Ehrlich 15	Ehrlich 32	Ehrlich 64
`E`	16	16	10	10	10
`C`	96	96	16	16	16
`Kc`	7	7	4	7	7
`Kx`	5	5	2	2	2
`Sx`	4	4	2	2	2
`H`	192	192	128	128	128

Table 2: CNN guide configuration for VSD and CbAS

$M = 64$**:** motif length = 4, no. motifs = 8, quantization = 4

D.3 GenBO settings

Acronym	Meaning
EI	Expected Improvement
PI	Probability of Improvement
sEI	Soft Expected Improvement, i.e., $\mathrm{softplus}(y - \tau)$
SR	Simple Regret (utility function)
fKL	Forward KL loss
bfKL	Balanced forward KL loss
rPL	Robust preference loss
MF	Mean-field categorical proposal model
Tfm	Transformer proposal model
fr	More frequent regularization (change in λ_n schedule rate)
r0p10	Base regularization factor set to $\lambda_0 := 0.1$
exp	Exponential regularizer, i.e., $R_n(\theta) := \lambda_n \exp\|\theta - \theta_0\|_2^2$
np	No (informative) prior, i.e., $p_0(\mathbf{x}) \propto 1$
p	Pre-trained prior, learned from initial (randomly initialized) data $\mathcal{D}_0$
lg	Importance weights
lr0p10	Learning rate setting for training the generative model (e.g., 0.1 in this case)
pcmin0p50	Minimum percentile for threshold τ_t annealing schedule (e.g., 50% in this case)
pcmax0p90	Maximum percentile for threshold τ_t annealing schedule (e.g., 90% in this case)

Table 3: GenBO experiment settings acronyms

Table 3 presents our settings for the different GenBO variants across experiments. The settings for our proposal models followed VSD's configurations. Our regularization scheme penalized the Euclidean distance between the model's parameters and their random initialization (He et al., 2015) with $R_n(\theta) := \lambda_n\|\theta - \theta_0\|_2^2$, using an annealed regularization factor $\lambda_n := \frac{1}{n}\lambda_0 \log^2 n$, similar to Dai et al. (2022), which ensures enough exploration, while still $\lambda_n \to 0$ as $n \to \infty$, allowing for convergence to the optimal θ_*. For threshold-based utilities, we mainly set the quantile threshold τ_t to follow an annealing schedule ranging from the 50% (i.e., the median) to the 99% percentile of the observations marginal distribution for both GenBO and VSD, where the percentile γ_t corresponding the quantile is updated as $\gamma_t := \gamma_{t-1}^{\eta}$, where $\eta := \left(\frac{\log \gamma_T}{\log \gamma_0}\right)^{\frac{1}{T-1}} \in (0, 1)$.

	ALOHA	Ehrlich-15	Ehrlich-32	Ehrlich-64
Random mut.	3.80 ± 0.40			
LaMBO-2		0.19 ± 0.17	0.36 ± 0.15	0.95 ± 0.02
CbAS	2.20 ± 0.40	0.57 ± 0.12	0.61 ± 0.10	0.98 ± 0.01
GA		0.45 ± 0.12	0.61 ± 0.10	0.98 ± 0.01
VSD	**0.00** ± **0.00**	0.19 ± 0.17	0.32 ± 0.09	0.97 ± 0.00
GenBO	0.20 ± 0.40	**0.00** ± **0.00**	**0.28** ± **0.16**	**0.94** ± **0.02**

Table 4: Final average regret (lower is better) for the best-performing variant of each method across the ALOHA (text optimization) and Ehrlich benchmarks

	FoldX (Stability)	FoldX (SASA)
Random mut.	2.79 ± 0.22	12550.26 ± 56.34
LaMBO-2	3.19 ± 0.58	12456.10 ± 126.64
CbAS	3.65 ± 0.23	12376.65 ± 298.30
VSD	**4.20** ± **0.42**	12537.97 ± 186.35
GenBO	3.28 ± 0.35	**13285.42** ± **221.60**

Table 5: FoldX average maximum outcome for the best-performing variant of each method

D.4 Results summary

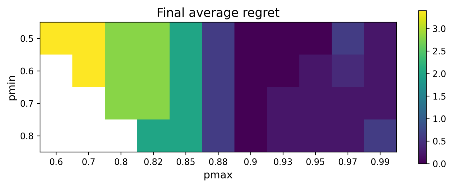
Besides the plots in section 6, we summarize the final results in Table 4 and 5.

D.5 Ablations

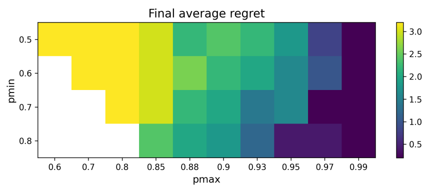
This section presents ablation studies. We vary the minimum and maximum percentile for the threshold annealing settings of both GenBO (with PI utility) and VSD on the text optimization problem in Figure 5. In Figure 6, we vary the batch size B for GenBO on the Ehrlich benchmark problem of sequence length 32.

Method	Average Runtime
CbAS	53.38 s ± 2.05 s
VSD	42.89 s ± 2.56 s
GenBO	**14.88 s ± 0.26 s**

Table 6: Summary of average run times with standard deviations on the `ALOHA` text optimization problem.



(a) GenBO



(b) VSD

Figure 5: Final average simple regret for GenBO and VSD as a function of the minimum and maximum percentile in the annealing schedule.

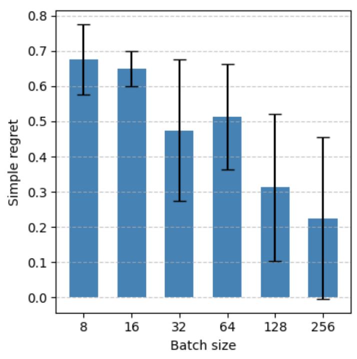


Figure 6: Batch evaluation size B ablation on Ehrlich benchmark of length 32. The plot presents the final average simple regret for each B setting, with error bars corresponding to ± 1 standard deviation. All variants were run for the same number of BO rounds as in the original experiment.