Posterior Inference in Latent Space for Scalable Constrained Black-box Optimization

Kiyoung Om[*]
UT Austin
se99an@utexas.edu

Kyuil Sim[*]
KAIST
kyuil.sim@kaist.ac.kr

Taeyoung Yun[*]
KAIST
99yty@kaist.ac.kr

Hyeongyu Kang
KAIST
khg2000v@kaist.ac.kr

Jinkyoo Park
KAIST, Omelet
jinkyoo.park@kaist.ac.kr

Abstract

Optimizing high-dimensional black-box functions under black-box constraints is a pervasive task in a wide range of scientific and engineering problems. These problems are typically harder than unconstrained problems due to hard-to-find feasible regions. In this work, we reformulate constrained black-box optimization as posterior inference, and perform this inference in the latent space of generative models. Our method iterates through two stages. First, we train flow-based models to capture the data distribution and surrogate models that predict both function values and constraint violations. Second, we cast the candidate selection problem as a posterior inference problem to effectively search for promising candidates that have high objective values while not violating the constraints. Concretely, we utilize outsourced diffusion models to amortize the sampling from the posterior distribution in the latent space of flow-based models, which can bypass the issue of mode collapse. We empirically demonstrate that our method achieves superior performance across synthetic and real-world tasks. Our code is available here.

1 Introduction

Optimizing high-dimensional black-box functions under black-box constraints is a fundamental task across numerous scientific and engineering problems, including machine learning [1], drug discovery [2], control [3], and industrial design [4]. These problems are substantially harder than their unconstrained counterparts. Feasible regions are analytically undefined and often occupy only a small fraction of the search space [5], and high dimensionality compounds all of these difficulties.

Bayesian optimization (BO) has been widely used to solve black-box optimization problems in a sample-efficient manner [6, 7]. Some BO methods extend to black-box constraints by developing new acquisition functions [1, 8] or relaxing them via Lagrangian methods [9, 10]. However, BO inherently scales poorly to high dimensionality even without constraints [11], and incorporating constraints further complicates the function landscape, hindering accurate estimation of surrogate models.

Generative models have recently emerged as a promising alternative for optimization problems with black-box constraints [12, 13, 14]. By capturing the distribution of promising candidates, they can explore the search space more flexibly than solving acquisition function maximization [15]. However, in high-dimensional constrained spaces, the target distribution over feasible and high-scoring candidates is highly multi-modal and exhibits large flat regions, making sampling from this complex distribution intractable. One can approximate it via Markov Chain Monte Carlo (MCMC),

[*]Equal contribution.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

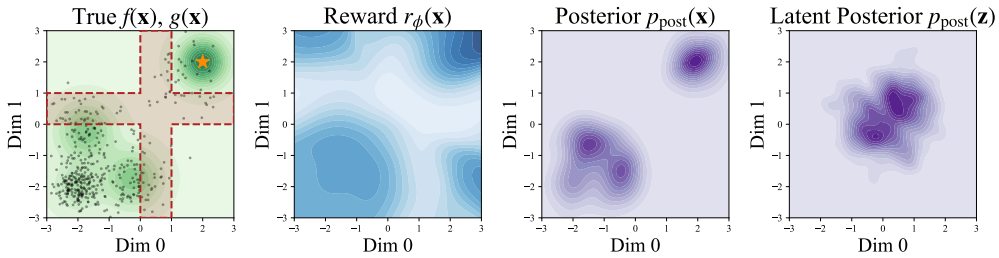


Figure 1: Motivating Figure (★: global optimum, - - -: infeasible region). In constrained optimization problems, the posterior distribution in the data space, $p_{\text{post}}(\mathbf{x})$, is highly multi-modal since the constraint penalty makes a complex reward landscape. In latent space, $p_{\text{post}}(\mathbf{z})$ is smoother and more amenable to amortized inference, enabling robust exploration towards promising regions.

but such methods scale poorly to high-dimensional spaces [16, 17]. Fine-tuning approaches to match the target distribution are prone to mode collapse when the distribution is highly multi-modal [18, 19].

In this paper, we propose **CiBO**, a novel framework for scalable constrained black-box optimization that overcomes the above limitations. We formalize candidate selection as sampling from the posterior distribution $p_{\text{post}}(\mathbf{x}) \propto p_\theta(\mathbf{x}) \exp(\beta \cdot r_\phi(\mathbf{x}))$, where p_θ is a learned flow-based prior and r_ϕ is a proxy that encodes both high objective values and constraint feasibility. Our key observation is that while the posterior has distinct modes in data space, the corresponding posterior in the latent space is significantly smoother, as illustrated in Figure 1. Therefore, we amortize posterior inference in the latent space and project latents into the data space rather than fine-tuning flow-based models.

Our method iterates through two stages. In **Phase 1**, we train a flow-based model on the current dataset via reweighting, alongside an ensemble of surrogate models with uncertainty quantification to predict objective values and constraints. In **Phase 2**, we sample candidates by training an outsourced diffusion sampler [19] that amortizes the posterior distribution in the latent space via the trajectory balance objective [20]. After sampling and filtering, we evaluate the selected candidates, update the dataset, and repeat until the evaluation budget is exhausted.

We conduct extensive experiments on three synthetic and four real-world benchmarks, including a more challenging indicator-constraint setting where only binary feasibility feedback is available. We empirically show that CiBO outperforms several competitive baselines from BO to generative model-based approaches across different tasks. Beyond performance, our ablation studies reveal that amortized inference in the latent space enables robust exploration on feasible regions.

2 Related Works

2.1 Constrained Black-box Optimization

Most scientific and engineering optimization problems involve black-box constraints, such as the synthesizability of molecules in chemical design [2] and safety constraints in robot control policies [3]. To handle such constraints, various strategies have been proposed in the BO literature. cEI [21] and LogcEI [22] incorporate constraints violations into the acquisition function. To scale to high-dimensional search spaces, SCBO [5] proposes a trust-region approach with bilog transformations to address scalable constrained black-box optimization problems. PCAGP-SCBO [4] extends this idea by applying PCA [23] to handle a large number of constraints. FuRBO [24] proposes a feasibility-driven trust region approach that adapts the search region based on constraint satisfaction.

Another line of work integrates the objective function with constraints via augmented Lagrangian or the Alternating Direction Method of Multipliers (ADMM) [9, 10], and solves unconstrained optimization problems by vanilla BO methods. While our method also applies Lagrangian relaxation, we take a fundamentally different approach from standard BO literature in the candidate selection procedure. Instead of searching for inputs that maximize the acquisition function, we frame candidate selection as a posterior inference problem.

Constrained black-box optimization has also been studied in evolutionary algorithms and operations research. CMA-ES [25, 26] handles constraints either by assigning the objective values of infeasible points to zero, while COBYLA[27] maintains trust regions to perform local search in feasible regions.

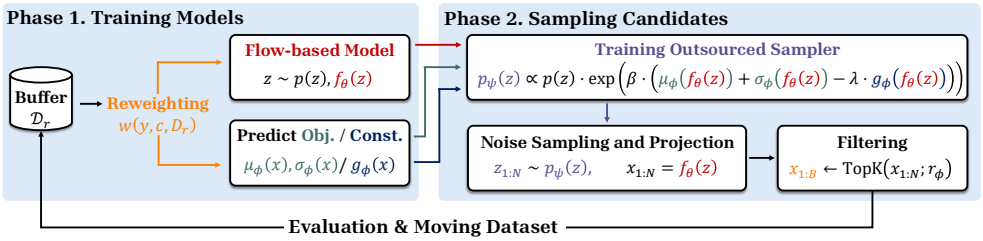


Figure 2: Overview of our method. **Phase 1:** Train flow-based models and proxies for the objective and constraints. **Phase 2:** Sample candidates from the posterior distribution using an outsourced diffusion sampler. After sampling, we utilize filtering to enhance sample efficiency. Then, we evaluate samples, update the dataset, and repeat the process until the evaluation budget is exhausted.

2.2 Generative Model-based Optimization

Motivated by the success of generative models [28, 29], several methods have been proposed to leverage generative models for black-box optimization, both with and without constraints.

In offline black-box optimization, DDOM [30] trains a conditional diffusion model with classifier-free guidance [31] and applies loss reweighting to emphasize samples with high objective values. DiffOPT [12] formulates offline optimization as a constrained optimization problem. It applies diffusion to capture data distribution, followed by Langevin dynamics or an iterative importance sampling procedure to explore promising regions while not deviating too far from the data distribution.

In online black-box optimization, DiffBBO [32] and DiBO [15] both leverage diffusion models for candidate selection, but are designed to solve unconstrained optimization problems. Unfortunately, utilizing generative models for constrained black-box optimization in the online setting remains unexplored. In this paper, we propose an effective generative model-based method that performs posterior inference in the latent space to effectively search for candidates in feasible regions.

2.3 Posterior Inference in Flow-based and Diffusion Models

Given a flow-based or diffusion prior $p_\theta(\mathbf{x})$ trained on a dataset and a reward function $r(\mathbf{x})$, sampling from the unnormalized posterior $p_{\text{post}}(\mathbf{x}) \propto p_\theta(\mathbf{x})r(\mathbf{x})$ has numerous applications in downstream tasks such as conditional image generation [31, 33], Bayesian inverse problems [16, 18], and aligning pretrained models with human preference [34, 19]. However, direct sampling from this unnormalized posterior distribution $p_{\text{post}}(\mathbf{x})$ is generally intractable [18, 35].

To address this problem, some approaches train classifiers directly within intermediate noised spaces [33, 36] while others approximate posterior sampling via Markov Chain Monte Carlo (MCMC) procedures [37, 16, 17, 38]. However, these methods struggle to scale to high-dimensional settings. Conversely, several methods use reinforcement learning [34, 39] or stochastic optimal control [40] to amortize posterior sampling. These methods primarily fine-tune the parameters of the prior using on-policy samples. Unfortunately, naive implementations of fine-tuning methods can be prone to mode collapse when the target distribution is highly multi-modal and has a large flat region [18, 19].

To mitigate this issue, we adopt the outsourced diffusion sampler method proposed by Venkatraman et al. [19]. In the outsourced diffusion sampler, we train an additional diffusion model that samples latents, which are then mapped into the data space via the deterministic mapping derived from the prior. Matching the distribution within the latent space significantly simplifies the alignment task when the distribution is highly multi-modal and has a large flat region in the original data space.

3 Preliminaries

3.1 Constrained Black-box Optimization

We consider the following constrained black-box optimization problem.

$$\text{find } \mathbf{x}^* = \arg\max_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) \quad \text{s.t. } g^{(1)}(\mathbf{x}) \leq 0, \cdots, g^{(M)}(\mathbf{x}) \leq 0 \tag{1}$$

The objective function $f : \mathcal{X} \to \mathbb{R}$ and constraints $g^{(1)}, \cdots, g^{(M)} : \mathcal{X} \to \mathbb{R}$ are black-box functions. We assume an evaluation budget of R rounds, each with a batch of B queries. We also consider a more challenging scenario in which the only available feedback is a binary indicator of whether each constraint is violated. We refer to this as an indicator constraint.

3.2 Flow-based Models

Flow-based models [41, 42, 43] are a class of generative models for approximating a target distribution $q(\mathbf{x})$. Flow-based models are defined via the deterministic ordinary differential equation (ODE).

$$d\mathbf{x}_t = v_\theta(\mathbf{x}_t, t)\, dt \quad v_\theta(\mathbf{x}_t, t) : \mathbb{R}^d \times [0, 1] \to \mathbb{R}^d \tag{2}$$

where v_θ is a parameterized velocity field.

For each given velocity field, the corresponding flow $\psi_\theta(\mathbf{x}_0, t) : \mathbb{R}^d \times [0, 1] \to \mathbb{R}^d$ satisfies:

$$\frac{d}{dt}\psi_\theta(\mathbf{x}_0, t) = v_\theta(\psi_\theta(\mathbf{x}_0, t), t), \quad \psi_\theta(\mathbf{x}_0, 0) = \mathbf{x}_0. \tag{3}$$

The velocity field $v_\theta(\mathbf{x}_t, t)$ generates a continuous probability path p_t induced by the flow:

$$\mathbf{x}_t = \psi_\theta(\mathbf{x}_0, t) \sim p_t, \quad \text{where} \quad \mathbf{x}_0 \sim p_0. \tag{4}$$

Training Flow-based Models. We use Flow Matching [41] to learn the v_θ that generates a path interpolating smoothly between an initial distribution $p_0 = p$ and a target distribution $p_1 = q$.

We employ the simplest linear interpolation path $\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\mathbf{x}_1$, with derivative $\frac{d\mathbf{x}_t}{dt} = \mathbf{x}_1 - \mathbf{x}_0$. The conditional Flow Matching loss takes the following form.

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{\mathbf{x}_0\sim\mathcal{N}(0,I),\ \mathbf{x}_1\sim q(\mathbf{x}),\ t\sim\text{Unif}(0,1)} \left[\|v_\theta(\mathbf{x}_t, t) - (\mathbf{x}_1 - \mathbf{x}_0)\|_2^2\right] \tag{5}$$

3.3 Posterior Inference in Flow-based and Diffusion Models

Given a pretrained flow-based prior $p_\theta(\mathbf{x})$ and a reward function $r(\mathbf{x})$, we often need to sample from the posterior distribution $p_{\text{post}}(\mathbf{x}) \propto p_\theta(\mathbf{x})r(\mathbf{x})$. However, direct sampling from this posterior is mostly intractable.

We introduce outsourced diffusion sampling [19] to address this intractability. In outsourced diffusion sampling, we decompose the sampling process of generative models into a noise generation step $\mathbf{z} \sim p(\mathbf{z})$, followed by a deterministic transformation $\mathbf{x} = f_\theta(\mathbf{z})$, where $p(\mathbf{z})$ is standard normal and f_θ represents the learned mapping derived from the prior. Under this formulation, by Proposition 3.1 of Venkatraman et al. [19], we can sample from the posterior distribution by substituting noise generation as $\mathbf{z} \sim p_{\text{post}}(\mathbf{z}) \propto p(\mathbf{z})r(f_\theta(\mathbf{z}))$. To approximate the target distribution $p_\psi(\mathbf{z}) \approx p_{\text{post}}(\mathbf{z})$, we learn the parameters of the diffusion sampler ψ by minimizing the TB objective [20].

$$\mathcal{L}_{\text{TB}}(\mathbf{z}_{0:1}; \psi) = \left(\log \frac{Z_\psi p(\mathbf{z}_0) \prod_{i=0}^{T-1} p_F(\mathbf{z}_{(i+1)\Delta t}|\mathbf{z}_{i\Delta t}; \psi)}{p(\mathbf{z}_1) r(f_\theta(\mathbf{z}_1)) \prod_{i=1}^{T} p_B(\mathbf{z}_{(i-1)\Delta t}|\mathbf{z}_{i\Delta t})}\right)^2, \tag{6}$$

where Z_ψ is the parameterized partition function estimator, $(\mathbf{z}_0 \to \mathbf{z}_{\Delta_t} \to \cdots \mathbf{z}_1 = \mathbf{z})$ is the discrete time Markov chain of reverse-time stochastic differential equation (SDE) with time increment $\Delta t = \frac{1}{T}$. p_F and p_B are transition kernels of the discretized reverse and forward SDE.

4 Method

We introduce **CiBO**, a new framework for scalable constrained black-box optimization that leverages generative models for candidate selection. Our method consists of two iterative phases. First, we train a flow-based model to capture the data distribution alongside surrogate models that predict objective values and constraints. We then train a diffusion sampler that amortizes the posterior distribution in the latent space of the flow-based model, and draw candidates by projecting sampled latents into data space via its deterministic mapping. After sampling, we evaluate candidates, update the dataset, and repeat until the evaluation budget is exhausted. Figure 2 illustrates the overview of our method.

4.1 Phase 1. Training Models

In each round r, we have a pre-collected dataset $\mathcal{D}_r = \{\mathbf{x}_i, y_i, \mathbf{c}_i\}_{i=1}^I$, where $y_i = f(\mathbf{x}_i)$, $\mathbf{c}_i = \{c_i^m | c_i^m = g^{(m)}(\mathbf{x}_i),\ \forall m = 1, \ldots, M\}$, and I is the number of data points collected so far.

Training Prior. As the search space is high-dimensional, it is beneficial to implicitly constrain the search space close to the current data distribution. To this end, we first train a prior model p_θ to capture the current data distribution.

Training Surrogates. We also train surrogate models to predict both objective values and constraints. Because we can access only a small number of data points in the vast search space, we need to properly quantify the uncertainty of their predictions. To this end, we train an ensemble of K proxies $f_{\phi_1}, \ldots, f_{\phi_K}$ for the objective, following the deep ensemble method of Lakshminarayanan et al. [44]. For the constraints, we train individual proxies $g_\phi^{(1)}, \ldots, g_\phi^{(M)}$ without ensembling, since using an ensemble for each of the M constraints would lead to prohibitive time complexity.

Reweighted Training. We introduce reweighted training [45, 46, 30, 47] to focus on promising data points with high objective values while not violating constraints. Following Tripp et al. [45], we assign a weight to each data point:

$$l(y, \mathbf{c}) = y - \lambda \sum_{m=1}^{M} \max(0, c^m), \quad w\left(y, \mathbf{c}, \mathcal{D}_r\right) = \frac{1}{\kappa |\mathcal{D}_r| + \text{rank}_{l, \mathcal{D}_r}(y, \mathbf{c})}, \tag{7}$$

where λ is a Lagrange multiplier, $\kappa \geq 0$ controls the strength of reweighting, and $\text{rank}_{l, \mathcal{D}_r}(y, \mathbf{c})$ is the rank of $(y, \mathbf{c})$ within $\mathcal{D}_r$ ordered by l. We then define the training objective for the proxies and the flow-based model via reweighting as follows:

$$\mathcal{L}(\phi) = \sum_{(\mathbf{x}, y, \mathbf{c}) \in \mathcal{D}_r} w(y, \mathbf{c}, \mathcal{D}_r) \cdot \left[\sum_{k=1}^{K} \left(y - f_{\phi_k}\left(\mathbf{x}\right)\right)^2 + \sum_{m=1}^{M} \left(c^m - g_\phi^{(m)}\left(\mathbf{x}\right)\right)^2 \right], \tag{8}$$

$$\mathcal{L}(\theta) = \mathbb{E}_{\mathbf{x}_0 \sim \mathcal{N}(0, I), (\mathbf{x}, y, \mathbf{c}) \in \mathcal{D}_r,\, t \sim \text{Unif}(0,1)} \left[w \| v_\theta(\mathbf{x}_t, t) - (\mathbf{x} - \mathbf{x}_0) \|_2^2 \right]. \tag{9}$$

4.2 Phase 2. Sampling Candidates

After training, we select candidates for evaluation in the next round. As the search space is high-dimensional, the prediction of surrogate models is likely to be inaccurate in regions that are too far away from the dataset collected so far. Therefore, it is advantageous to sample candidates from the distribution that satisfies the two desiderata: (1) promote exploration towards high-scoring and feasible regions, and (2) prevent sampling candidates that deviate too far from the current data distribution. To accomplish these objectives, we cast candidate selection as sampling from the following target distribution p_{post}.

$$p_{\text{post}}(\mathbf{x}) = \arg\max_{p \in \mathcal{P}} \mathbb{E}_{\mathbf{x} \sim p} \left[r_\phi(\mathbf{x}) \right] - \frac{1}{\beta} \cdot D_{\text{KL}} \left(p \,\|\, p_\theta \right), \tag{10}$$

where $\mathcal{P}$ is the space of all probability distributions over the domain $\mathcal{X}$, and

$$r_\phi(\mathbf{x}) = \mu_\phi(\mathbf{x}) + \gamma \cdot \sigma_\phi(\mathbf{x}) - \lambda \sum_{m=1}^{M} \max(0, g_\phi^{(m)}(\mathbf{x})). \tag{11}$$

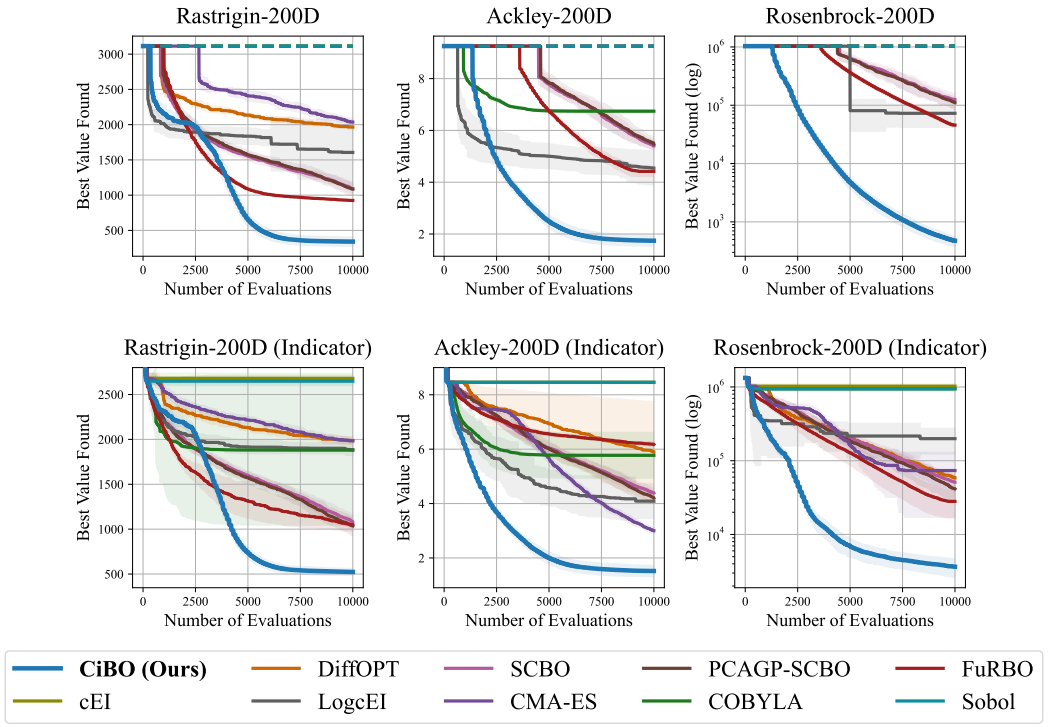


Figure 3: Comparison of CiBO with baselines on synthetic tasks. Experiments used 12 random seeds, reporting the mean and one standard deviation. A dashed line indicates no feasible solutions found.

$\mu_\phi(\mathbf{x})$ and $\sigma_\phi(\mathbf{x})$ represent the mean and standard deviation from the ensemble of surrogate models for the objective. β is an inverse temperature, and λ is a Lagrange multiplier.

Following the derivation in Nair et al. [48], the target distribution admits the closed-form expression

$$p_{\text{post}}(\mathbf{x}) \propto p_\theta(\mathbf{x}) \exp\left(\beta \cdot [r_\phi(\mathbf{x})]\right). \tag{12}$$

If we treat the flow-based model $p_\theta(\mathbf{x})$ as a prior and the exponential term $\exp(\beta \cdot [r_\phi(\mathbf{x})])$ as a reward $r(\mathbf{x})$, then our objective is to sample from the posterior distribution.

Amortized Inference in Latent Space. As introduced in Section 3.3, we can view the sampling procedure of flow-based models as drawing samples from the standard normal distribution $\mathbf{z} \sim p(\mathbf{z})$, followed by the deterministic transformation $\mathbf{x} = f_\theta(\mathbf{z})$, where f_θ is a deterministic mapping derived from the pretrained flow-based model. Within this framework, we can generate samples from $p_{\text{post}}(\mathbf{x})$ by modifying the latent generation distribution.

$$\mathbf{z} \sim p_{\text{post}}(\mathbf{z}) \propto p(\mathbf{z}) r(f_\theta(\mathbf{z})). \tag{13}$$

To sample latents $\mathbf{z}$ from the latent-space posterior $p_{\text{post}}(\mathbf{z})$, we train a diffusion model $p_\psi(\mathbf{z})$ to amortize $p_{\text{post}}(\mathbf{z})$ by minimizing the TB objective.

$$\mathcal{L}_{\text{TB}}(\mathbf{z}_{0:1}; \psi) = \left(\log \frac{Z_\psi p(\mathbf{z}_0) \prod_{i=0}^{T-1} p_F(\mathbf{z}_{(i+1)\Delta t} | \mathbf{z}_{i\Delta t}; \psi)}{p(\mathbf{z}_1) r(f_\theta(\mathbf{z}_1)) \prod_{i=1}^{T} p_B(\mathbf{z}_{(i-1)\Delta t} | \mathbf{z}_{i\Delta t})} \right)^2. \tag{14}$$

4.3 Filtering, Evaluation and Moving Dataset

Filtering. After sampling from the posterior distribution, we carefully select candidates to improve sample efficiency of our method. We generate $N \cdot B$ samples from the amortized sampler and select the top-B samples based on Equation (11) as candidates.

Evaluation and Dataset Management. We evaluate the objective function and constraint functions for each selected candidate. Then, we update the dataset with new observations. We empirically find that taking only a subset of total observations is beneficial in terms of both time complexity and sample efficiency. Therefore, we remove the samples with the lowest score if the size of the dataset is larger than the buffer size L. We present the pseudocode of our method in Appendix A.

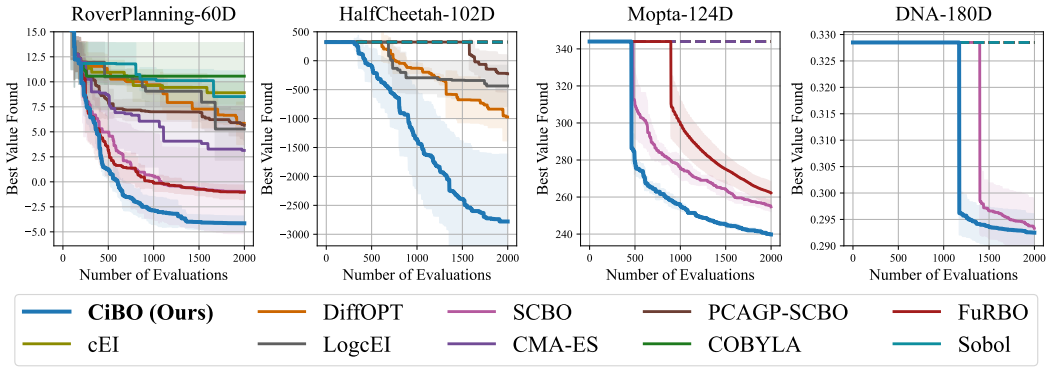


Figure 4: Comparison of CiBO with baselines on real world tasks. Experiments used 4 random seeds, reporting mean and one standard deviation. A dashed line indicates no feasible solutions found.

5 Experiments

We evaluate CiBO on three 200-dimensional synthetic functions and four real-world benchmarks spanning 60 to 180 dimensions. For the synthetic tasks, we also consider a more challenging indicator constraint setting where only binary feasibility feedback is available. For all tasks, we report the best feasible value found and assign the worst observed value across all methods to infeasible runs, following Hern et al. [49], Eriksson and Poloczek [5].

5.1 Tasks

We consider three synthetic and four real-world benchmarks for evaluating our method.

- **Synthetic.** Rastrigin ($D = 200$), Ackley ($D = 200$), and Rosenbrock ($D = 200$), each subject to two inequality constraints: $\sum_{d=1}^{200} x_d \leq 0$ and $\|\mathbf{x}\|_2^2 \leq 30$. We also consider a more challenging indicator constraint setting where only binary feasibility feedback is available.
- **Real-world.** RoverPlanning ($D = 60$, 15 constraints), HalfCheetah ($D = 102$, 5 constraints), Mopta ($D = 124$, 68 constraints), and Lasso DNA ($D = 180$, 5 constraints).

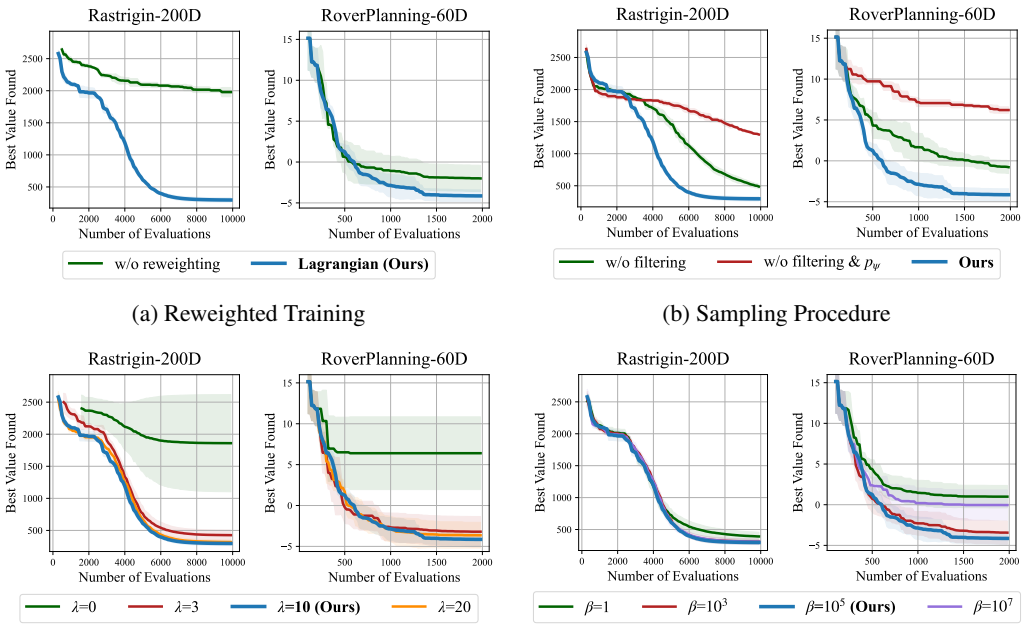
For synthetic tasks, we use $|\mathcal{D}_0| = 200$, batch size $B = 100$, and a budget of 10,000 evaluations. For the indicator constraint setting, we seed the initial dataset with 10 feasible points obtained via hit-and-run MCMC, as finding an initial feasible solution is otherwise too challenging for all baselines. For real-world tasks, we use $|\mathcal{D}_0| = 200$, a budget of 2,000 evaluations, and batch size $B = 50$ for all tasks except Mopta ($B = 20$), as baselines could not find feasible solutions with $B = 50$. Please refer to Appendix B for detailed task descriptions.

5.2 Baselines

We compare with constrained expected improvement methods cEI [21] and LogcEI [22], which extend standard BO acquisition functions with constraint handling. We also evaluate SCBO [5], PCAGP-SCBO [4], and FuRBO [24], which are specifically designed for scalable constrained black-box optimization. We further compare with DiffOPT [12], a generative model-based approach designed for constrained optimization. We also include the evolutionary search algorithm CMA-ES [25], the derivative-free optimizer COBYLA [27], and the quasi-random Sobol sequences [50]. Detailed settings are provided in Appendix C.

5.3 Implementation

We parametrize the surrogate models, the flow-based model, and the diffusion sampler all with lightweight MLPs. At every optimization round, we retrain the proxies for 100 steps, the flow model for 500 steps, and the diffusion sampler for 100 steps. As shown in the runtime breakdown in Appendix F, this lightweight design keeps the per-round computational cost comparable to or lower than that of BO-based approaches. Please refer to Appendix D for implementation details.



(a) Reweighted Training

(b) Sampling Procedure

(c) Analysis on λ

(d) Analysis on β

Figure 5: Additional analysis for various components of CiBO. Experiments are conducted with four random seeds, and the mean and one standard deviation are reported.

5.4 Synthetic Experiments

As shown in Figure 3, CiBO outperforms all baselines on all three tasks under both constraint settings. Among BO baselines, SCBO, PCAGP-SCBO, LogcEI, and FuRBO identify feasible solutions but plateau early due to the limited expressiveness of GP surrogates in 200 dimensions. cEI, CMA-ES, and COBYLA fail to find feasible solutions on some tasks, and DiffOPT struggles with low sample efficiency across several tasks. These results show that posterior inference in latent space enables CiBO to find feasible solutions and maximize the objective in a sample-efficient manner.

5.5 Real World Experiments

As shown in Figure 4, CiBO consistently identifies high-quality feasible solutions with superior sample efficiency across all tasks, and the performance gap relative to competing baselines is substantially more pronounced than in the synthetic tasks. DiffOPT achieves feasibility only on HalfCheetah, failing to identify feasible solutions on Mopta-124D and DNA-180D. FuRBO recovers feasible solutions on Mopta, which involves a large number of constraints, yet exhibits limited overall performance. SCBO is the sole competing method to achieve feasibility across most tasks but with low sample efficiency, further highlighting the robustness of CiBO in scalable constrained settings.

5.6 Additional Analysis

In this section, we conduct an analysis of each component of our proposed method through ablation experiments on Rastrigin 200D and Rover Planning 60D tasks.

Feasibility Ratio. We also conduct feasibility-ratio analysis of the proposed candidates of CiBO. As depicted in Appendix E.1, CiBO reaches near-perfect feasibility within 5$\sim$10 batches on Rastrigin-200D, while BO baselines require roughly twice as many batches to reach comparable ratios. On RoverPlanning-60D, most baselines remain at a low feasibility ratio, whereas CiBO maintains steady improvement throughout the optimization.

Reweighted Training. To investigate the effectiveness of the reweighted training approach in Equation (7), we compare against a variant trained without reweighting (i.e., uniform sampling). As shown in Figure 5a, removing reweighting significantly reduces sample efficiency, indicating that it

effectively handles objectives and feasibility in high-dimensional space. We further compare different reweighting strategies (e.g., score-weighted vs. rank-based) in Appendix E.2.

Sampling Procedure. To analyze the effect of each component during candidate sampling, we experiment with two variants: removing filtering, and removing both filtering and the diffusion sampler, thus sampling candidates directly from the prior p_θ. As shown in Figure 5b, a significant gap exists between ours and the other variants, validating the effectiveness of each proposed component. We also employ off-policy training for the amortized diffusion sampler to improve mode coverage of the posterior distribution. An ablation on this component is provided in Appendix E.3.

Lagrange Multiplier λ. To assess the impact of penalty strength, we experiment with different values of the λ. As shown in Figure 5c, setting $\lambda = 0$ significantly degrades performance on both tasks, as the method then neglects feasibility. Conversely, our method shows robustness to the specific choice of $\lambda > 0$, consistently improving performance. We also compare against an adaptive λ schedule in Appendix E.4 and find that it adds complexity without improving performance.

Inverse Temperature β. The inverse temperature controls the balance between the prior $p_\theta(\mathbf{x})$ and the reward function $r(\mathbf{x})$. While we fix $\beta = 10^5$ across all experiments, we conduct ablation studies on the controllability of β in Figure 5d. A moderately high β improves sample efficiency, but if β is too high, performance becomes sensitive to surrogate model accuracy, leading to sub-optimal results. Too small β leads to slow convergence due to lack of guidance toward high-scoring regions.

Buffer size L. For each round, we maintain a subset of the collected dataset with a buffer size L. We set $L = 2000$ for synthetic tasks and $L = 1000$ for real-world tasks, as the evaluation budget for real-world tasks is 2000. We analyze the effect of L in Appendix E.5 and find that using a small L leads to early saturation and sub-optimal solutions, while a large L slows the rate of improvement.

Initial Dataset size $|\mathcal{D}_0|$ and Batch size B. The initial dataset size $|\mathcal{D}_0|$ and batch size B are important design choices that can significantly affect the performance of optimization methods. A small $|\mathcal{D}_0|$ may provide insufficient coverage of the search space, while a large B increases the per-round evaluation cost. We analyze sensitivity to both in Appendix E.6 and find that CiBO maintains consistent performance across different configurations, demonstrating robustness to these choices.

Misaligned Constraints. In the synthetic tasks, the optimal points of Rastrigin and Ackley coincide with the center of the norm constraint, which may artificially favor all methods. To verify that CiBO does not rely on this alignment, we test on both Rastrigin-200D and Ackley-200D with shifted constraint centers. As shown in Appendix E.7, CiBO consistently outperforms the baselines on both tasks, confirming that its advantage stems from expressive posterior inference rather than constraint-objective alignment.

Comparison with Unconstrained Generative Methods. DDOM [30] and DiBO [15] are generative model-based methods designed for unconstrained optimization. We adapt both methods to the constrained setting via Lagrangian relaxation with independently searched λ and provide a comparison in Appendix E.8. As shown in the results, both methods underperform CiBO, highlighting the importance of explicitly incorporating constraint handling into the framework.

Scalability. We report a per-round runtime breakdown in Appendix F, decomposing each round into surrogate fitting, generative model training, candidate proposal, and oracle evaluation. Even with several components to train, CiBO exhibits a low runtime per round as it uses lightweight models for both surrogate and generative stages, avoiding the expensive GP fitting in BO-based methods.

6 Conclusion

We introduced CiBO, a generative model-based framework for scalable constrained black-box optimization. Our approach formulates candidate selection as posterior inference, leveraging flow-based models to capture the data distribution and surrogate models to predict both objectives and constraints. By amortizing posterior sampling in the latent space with outsourced diffusion samplers, our method effectively addresses the challenges posed by highly multi-modal and flat posterior

distributions that arise from incorporating constraints. Extensive experiments across synthetic and real-world benchmarks demonstrate the superiority of our proposed method.

Limitations and Future Works. We are interested in improving our method. First, as we need to train all models with the updated dataset in every round, presenting a framework that can reuse the trained models from the previous rounds would be beneficial. Furthermore, there are several advancements in the literature on flow-based model training [51] and diffusion samplers [52, 53], which could potentially yield substantial performance gains. We leave them as future work.

7 Acknowledgement

We thank the anonymous reviewers for their insightful comments and suggestions, which significantly improve our manuscript. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2024-00410082).

References

[1] Jacob Gardner, Matt Kusner, Kilian Weinberger, John Cunningham, et al. Bayesian optimization with inequality constraints. In *International Conference on Machine Learning (ICML)*, 2014.

[2] Ryan-Rhys Griffiths and José Miguel Hernández-Lobato. Constrained bayesian optimization for automatic chemical design using variational autoencoders. *Chemical science*, 2020.

[3] Felix Berkenkamp, Angela P Schoellig, and Andreas Krause. Safe controller optimization for quadrotors with gaussian processes. In *International Conference of Robotics and Automation (ICRA)*, 2016.

[4] Hauke F Maathuis, Roeland De Breuker, and Saullo GP Castro. High-dimensional bayesian optimisation with large-scale constraints via latent space gaussian processes. *arXiv preprint arXiv:2412.15679*, 2024.

[5] David Eriksson and Matthias Poloczek. Scalable constrained bayesian optimization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2021.

[6] HJ Kushner. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal of Basic Engineering*, 1964.

[7] Roman Garnett. *Bayesian optimization*. Cambridge University Press, 2023.

[8] José Miguel Hernández-Lobato, Michael Gelbart, Matthew Hoffman, Ryan Adams, and Zoubin Ghahramani. Predictive entropy search for bayesian optimization with unknown constraints. In *International Conference on Machine Learning (ICML)*, 2015.

[9] Victor Picheny, Robert B Gramacy, Stefan Wild, and Sebastien Le Digabel. Bayesian optimization under mixed constraints with a slack-variable augmented lagrangian. *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[10] Setareh Ariafar, Jaume Coll-Font, Dana Brooks, and Jennifer Dy. Admmbo: Bayesian optimization with unknown constraints using admm. *Journal of Machine Learning Research (JMLR)*, 2019.

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

[12] Lingkai Kong, Yuanqi Du, Wenhao Mu, Kirill Neklyudov, Valentin De Bortoli, Dongxia Wu, Haorui Wang, Aaron M Ferber, Yian Ma, Carla P Gomes, and Chao Zhang. Diffusion models as constrained samplers for optimization with unknown constraints. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2025.

[13] Wenqian Xing, JungHo Lee, Chong Liu, and Shixiang Zhu. Black-box optimization with implicit constraints for public policy. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2025.

[14] Masatoshi Uehara, Xingyu Su, Yulai Zhao, Xiner Li, Aviv Regev, Shuiwang Ji, Sergey Levine, and Tommaso Biancalani. Reward-guided iterative refinement in diffusion models at test-time with applications to protein and dna design. In *International Conference on Machine Learning (ICML)*, 2025.

[15] Taeyoung Yun, Kiyoung Om, Jaewoo Lee, Sujin Yun, and Jinkyoo Park. Posterior inference with diffusion models for high-dimensional black-box optimization. In *International Conference on Machine Learning (ICML)*, 2025.

[16] Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *International Conference on Learning Representations (ICLR)*, 2023.

[17] Luhuan Wu, Brian Trippe, Christian Naesseth, David Blei, and John P Cunningham. Practical and asymptotically exact conditional sampling in diffusion models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[18] Siddarth Venkatraman, Moksh Jain, Luca Scimeca, Minsu Kim, Marcin Sendera, Mohsin Hasan, Luke Rowe, Sarthak Mittal, Pablo Lemos, Emmanuel Bengio, Alexandre Adam, Jarrid Rector-Brooks, Yoshua Bengio, Glen Berseth, and Nikolay Malkin. Amortizing intractable inference in diffusion models for vision, language, and control. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[19] Siddarth Venkatraman, Mohsin Hasan, Minsu Kim, Luca Scimeca, Marcin Sendera, Yoshua Bengio, Glen Berseth, and Nikolay Malkin. Outsourced diffusion sampling: Efficient posterior inference in latent spaces of generative models. In *International Conference on Machine Learning (ICML)*, 2025.

[20] Nikolay Malkin, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. Trajectory balance: Improved credit assignment in gflownets. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[21] Matthias Schonlau, William J Welch, and Donald R Jones. Global versus local search in constrained optimization of computer models. *Lecture notes-monograph series*, 1998.

[22] Sebastian Ament, Samuel Daulton, David Eriksson, Maximilian Balandat, and Eytan Bakshy. Unexpected improvements to expected improvement for bayesian optimization. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[23] Ian T Jolliffe. *Principal component analysis for special types of data*. Springer, 2002.

[24] Paolo Ascia, Elena Raponi, Thomas Bäck, and Fabian Duddeck. Feasibility-driven trust region bayesian optimization. In *International Conference on Automated Machine Learning (AutoML)*, 2025.

[25] Nikolaus Hansen. The cma evolution strategy: a comparing review. *Towards a new evolutionary computation: Advances in the estimation of distribution algorithms*, 2006.

[26] Asma Atamna, Anne Auger, and Nikolaus Hansen. Augmented lagrangian constraint handling for cma-es—case of a single linear constraint. In *International Conference on Parallel Problem Solving from Nature*, 2016.

[27] Michael JD Powell. A direct search optimization method that models the objective and constraint functions by linear interpolation. In *Advances in optimization and numerical analysis*. Springer, 1994.

[28] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

[29] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *International Conference on Machine Learning (ICML)*, 2024.

[30] Siddarth Krishnamoorthy, Satvik Mehul Mashkaria, and Aditya Grover. Diffusion models for black-box optimization. In *International Conference on Machine Learning (ICML)*, 2023.

[31] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS Workshop on Deep Generative Models and Downstream Applications*, 2021.

[32] Dongxia Wu, Nikki Lijing Kuang, Ruijia Niu, Yian Ma, and Rose Yu. Diff-BBO: Diffusion-based inverse modeling for black-box optimization. In *NeurIPS 2024 Workshop on Bayesian Decision-making and Uncertainty*, 2024.

[33] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[34] Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. Dpok: Reinforcement learning for fine-tuning text-to-image diffusion models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[35] Ruiqi Feng, Chenglei Yu, Wenhao Deng, Peiyan Hu, and Tailin Wu. On the guidance of flow matching. In *International Conference on Machine Learning (ICML)*, 2025.

[36] Cheng Lu, Huayu Chen, Jianfei Chen, Hang Su, Chongxuan Li, and Jun Zhu. Contrastive energy prediction for exact energy-guided diffusion sampling in offline reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2023.

[37] Florentin Coeurdoux, Nicolas Dobigeon, and Pierre Chainais. Normalizing flow sampling with langevin dynamics in the latent space. *arXiv preprint arXiv:2305.12149*, 2023.

[38] Gabriel Cardoso, Sylvain Le Corff, Eric Moulines, et al. Monte carlo guided denoising diffusion models for bayesian linear inverse problems. In *International Conference on Learning Representations (ICLR)*, 2024.

[39] Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2024.

[40] Carles Domingo-Enrich, Michal Drozdzal, Brian Karrer, and Ricky T. Q. Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. In *International Conference on Learning Representations (ICLR)*, 2025.

[41] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023.

[42] Xingchao Liu, Chengyue Gong, et al. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations (ICLR)*, 2023.

[43] Michael Samuel Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *International Conference on Learning Representations (ICLR)*, 2023.

[44] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[45] Austin Tripp, Erik Daxberger, and José Miguel Hernández-Lobato. Sample-efficient optimization in the latent space of deep generative models via weighted retraining. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[46] Aviral Kumar and Sergey Levine. Model inversion networks for model-based optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[47] Minsu Kim, Federico Berto, Sungsoo Ahn, and Jinkyoo Park. Bootstrapped training of score-conditioned generator for offline design of biological sequences. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[48] Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

[49] José Miguel Hern, Michael A Gelbart, Ryan P Adams, Matthew W Hoffman, Zoubin Ghahramani, et al. A general framework for constrained bayesian optimization using information-based search. *Journal of Machine Learning Research (JMLR)*, 2016.

[50] Ilya M Sobol. Distribution of points in a cube and approximate evaluation of integrals. *USSR Computational mathematics and mathematical physics*, 1967.

[51] Alexander Tong, Kilian FATRAS, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions of Machine Learning Research (TMLR)*, 2024.

[52] Aaron J Havens, Benjamin Kurt Miller, Bing Yan, Carles Domingo-Enrich, Anuroop Sriram, Daniel S Levine, Brandon M Wood, Bin Hu, Brandon Amos, Brian Karrer, et al. Adjoint sampling: Highly scalable diffusion samplers via adjoint matching. In *International Conference on Machine Learning (ICML)*, 2025.

[53] Minsu Kim, Sanghyeok Choi, Taeyoung Yun, Emmanuel Bengio, Leo Feng, Jarrid Rector-Brooks, Sungsoo Ahn, Jinkyoo Park, Nikolay Malkin, and Yoshua Bengio. Adaptive teachers for amortized samplers. In *International Conference on Learning Representations (ICLR)*, 2025.

[54] Nicola Demo, Marco Tezzele, and Gianluigi Rozza. A supervised learning approach involving active subspaces for an efficient genetic algorithm in high-dimensional optimization problems. *SIAM Journal on Scientific Computing*, 2021.

[55] Linnan Wang, Rodrigo Fonseca, and Yuandong Tian. Learning search space partition for black-box optimization using monte carlo tree search. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[56] Zeji Yi, Yunyue Wei, Chu Xin Cheng, Kaibo He, and Yanan Sui. Improving sample efficiency of high dimensional bayesian optimization with mcmc. In *6th Annual Learning for Dynamics & Control Conference*, 2024.

[57] Zelda B Zabinsky and Robert L Smith. Hit-and-run methods. *Encyclopedia of Operations Research and Management Science*, 2013.

[58] Zi Wang, Clement Gehring, Pushmeet Kohli, and Stefanie Jegelka. Batched large-scale bayesian optimization in high-dimensional spaces. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2018.

[59] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, 2012.

[60] MF Anjos and DR Jones. Mopta 2008 benchmark. *URL http://www. miguelanjos. com/jones-benchmark*, 2009.

[61] Leonard Papenmeier, Luigi Nardi, and Matthias Poloczek. Increasing the scope as you learn: Adaptive bayesian optimization in nested subspaces. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[62] Kenan Šehić, Alexandre Gramfort, Joseph Salmon, and Luigi Nardi. Lassobench: A high-dimensional hyperparameter optimization benchmark suite for lasso. In *International Conference on Automated Machine Learning (AutoML)*, 2022.

[63] Nikolaus Hansen, Youhei Akimoto, and Petr Baudis. CMA-ES/pycma on Github. Zenodo, DOI:10.5281/zenodo.2559634, February 2019. URL `https://doi.org/10.5281/zenodo.2559634`.

[64] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[65] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

[66] Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky TQ Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code. *arXiv preprint arXiv:2412.06264*, 2024.

[67] Ricky T. Q. Chen. torchdiffeq, 2018. URL `https://github.com/rtqichen/torchdiffeq`.

[68] Ulf Grenander and Michael I Miller. Representations of knowledge in complex systems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 1994.

[69] Simon Duane, Anthony D Kennedy, Brian J Pendleton, and Duncan Roweth. Hybrid monte carlo. *Physics letters B*, 1987.

[70] J. H. Halton. Sequential monte carlo. *Mathematical Proceedings of the Cambridge Philosophical Society*, 1962.

[71] Nicolas Chopin. A sequential particle filter method for static models. *Biometrika*, 2002.

[72] John Skilling. Nested sampling for general bayesian computation. 2006.

[73] Pablo Lemos, Nikolay Malkin, Will Handley, Yoshua Bengio, Yashar Hezaveh, and Laurence Perreault-Levasseur. Improving gradient-guided nested sampling for posterior inference. In *International Conference on Machine Learning (ICML)*, 2024.

[74] Qinsheng Zhang and Yongxin Chen. Path integral sampler: A stochastic control approach for sampling. In *International Conference on Learning Representations (ICLR)*, 2022.

[75] Francisco Vargas, Will Sussman Grathwohl, and Arnaud Doucet. Denoising diffusion samplers. In *International Conference on Learning Representations (ICLR)*, 2023.

[76] Julius Berner, Lorenz Richter, and Karen Ullrich. An optimal control perspective on diffusion-based generative modeling. *Transactions of Machine Learning Research (TMLR)*, 2024.

[77] Lorenz Richter and Julius Berner. Improved sampling via learned diffusions. In *International Conference on Learning Representations (ICLR)*, 2024.

[78] Francisco Vargas, Shreyas Padhy, Denis Blessing, and Nikolas Nüsken. Transport meets variational inference: Controlled monte carlo diffusions. In *International Conference on Learning Representations (ICLR)*, 2024.

[79] Salem Lahlou, Tristan Deleu, Pablo Lemos, Dinghuai Zhang, Alexandra Volokhova, Alex Hernández-Garcıa, Léna Néhale Ezzine, Yoshua Bengio, and Nikolay Malkin. A theory of continuous generative flow networks. In *International Conference on Machine Learning (ICML)*, 2023.

[80] Dinghuai Zhang, Ricky TQ Chen, Cheng-Hao Liu, Aaron Courville, and Yoshua Bengio. Diffusion generative flow samplers: Improving learning signals through partial trajectory optimization. In *International Conference on Learning Representations (ICLR)*, 2024.

[81] Marcin Sendera, Minsu Kim, Sarthak Mittal, Pablo Lemos, Luca Scimeca, Jarrid Rector-Brooks, Alexandre Adam, Yoshua Bengio, and Nikolay Malkin. Improved off-policy training of diffusion samplers. *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[82] Nikolay Malkin, Salem Lahlou, Tristan Deleu, Xu Ji, Edward J Hu, Katie E Everett, Dinghuai Zhang, and Yoshua Bengio. GFlownets and variational inference. In *International Conference on Learning Representations (ICLR)*, 2023.

Appendix

A Algorithms

Algorithm 1 CiBO

Input: Initial dataset $\mathcal{D}_0$; Max rounds R; Batch size B; Buffer size L; Number of constraints M; Flow model p_θ; Diffusion sampler p_ψ; Proxies $f_{\phi_1}, \cdots, f_{\phi_K}, g_\phi^{(1)}, \cdots, g_\phi^{(M)}$;
for $r = 0, \ldots, R-1$ **do**
  Initialize $p_\theta, p_\psi, f_{\phi_1}, \cdots, f_{\phi_K}, g_\phi^{(1)}, \cdots, g_\phi^{(M)}$

  Phase 1. Training Models
  Compute weights $w(y, \mathbf{c}, \mathcal{D}_r)$ with Equation (7)
  Train p_θ with Equation (9)
  Train $f_{\phi_1}, \cdots, f_{\phi_K}, g_\phi^{(1)}, \cdots, g_\phi^{(M)}$ with Equation (8)

  Phase 2. Sampling Candidates
  Train p_ψ with Equation (14) using prior p_θ and $\mathbf{z} \sim N(0, \mathbf{I})$
  Sample latent noise with $\{\mathbf{z}_i\}_{i=1}^{NB} \sim p_\psi(\mathbf{z})$
  Projection to data space with learned mapping $\mathbf{x}_i = f_\theta(\mathbf{z}_i) \quad \forall i \in \{1, \cdots, NB\}$

  Filtering
  Select top-B samples $\{\mathbf{x}_b\}_{b=1}^{B}$ with respect to $r_\phi(\mathbf{x})$

  Evaluation and Moving Dataset
  Evaluate $y_b = f(\mathbf{x}_b), \quad c_b^m = g^{(m)}(\mathbf{x}_b) \quad \forall m = \{1, \cdots M\} \quad \forall b = \{1, \cdots, B\}$
  Update $\mathcal{D}_{r+1} \leftarrow \mathcal{D}_r \cup \{(\mathbf{x}_b, y_b, \mathbf{c}_b)\}_{b=1}^{B}$
  if $|\mathcal{D}_{r+1}| > L$ **then**
    Remove last $|\mathcal{D}_{r+1}| - L$ samples from $\mathcal{D}_{r+1}$ with respect to: $y - \lambda \sum_{m=1}^{M} \max(0, c^m)$
  end if
end for

B Task Details

B.1 Synthetic Functions

We evaluate three synthetic functions in our constrained black-box optimization experiments: Rastrigin, Ackley, and Rosenbrock. The Rastrigin and Ackley functions are highly multi-modal functions with numerous local minima, whereas the Rosenbrock function features a narrow valley that makes convergence to the global minimum notoriously difficult [54]. Following [55, 56], we define the search domains as Rastrigin: $[-5, 5]^D$, Ackley: $[-5, 10]^D$, and Rosenbrock: $[-5, 10]^D$. All functions are subject to two constraints:

$$\sum_{d=1}^{200} x_d \leq 0 \quad \text{and} \quad ||\mathbf{x}||_2^2 \leq 30$$

Although prior work enforced the tighter bound $||\mathbf{x}||_2^2 \leq 5$, we relax this constraint in our high-dimensional setting. For the indicator constraint experiments, we sample initial feasible points by hit-and-run MCMC [57].

B.2 Rover Trajectory Planning

Rover Trajectory Planning is a trajectory optimization task in a 2D environment introduced by Wang et al. [58]. The objective is to optimize the rover's trajectory, where its trajectory is represented by 30 points defining a B-Spline. We place 15 impassable obstacles o_i and impose collision-avoidance constraints $c_i(\mathbf{x})$ as in Eriksson and Poloczek [5]:

$$c_i(\mathbf{x}) = \begin{cases} -\,d\big(o_i, \gamma(\mathbf{x})\big) & \text{if } \gamma(\mathbf{x}) \cap o_i = \varnothing, \\ \max\limits_{\alpha \in \gamma(\mathbf{x}) \cap o_i} \; \min\limits_{\beta \in \partial o_i} d(\alpha, \beta) & \text{otherwise.} \end{cases}$$

where $\gamma(\mathbf{x})$ denotes final trajectory, o_i is the region of the obstacle and ∂o_i denotes the boundary of o_i. A trajectory is feasible if and only if it does not intersect any obstacle. We follow the implementation from Wang et al. [58], but since there is no released code for the constraints, we implement the violation metric ourselves. Below is an example of the trajectory found by our method.

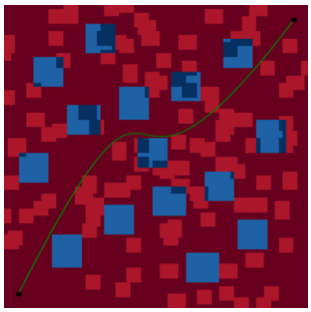


Figure 6: Trajectory found by CiBO, achieving minimum value of -4.14.

B.3 MuJoCo locomotion

The MuJoCo locomotion task [59] is a widely used benchmark in Reinforcement Learning. In this context, we aim to optimize a linear policy represented by the equation $\mathbf{a} = \mathbf{W}\mathbf{s}$, where $\mathbf{a}$ is the action and $\mathbf{s}$ is the state. Our objective is to maximize the average return of this policy by identifying the optimal weight matrix $\mathbf{W}$. We specifically focus on the HalfCheetah-102D. Each entry of the weight matrix $\mathbf{W}$ is constrained to the range $[-1, 1]$, and we conduct 5 rollouts for each evaluation. To reformulate the original problem for constrained optimization, we ensure that the return from each rollout is greater than -500. This setup represents a high-dimensional modification of the Lunar Landing task as discussed in Eriksson and Poloczek [5].

B.4 Vehicle Design with 68 Constraints (MOPTA)

MOPTA is the high-dimensional real-world problem of large-scale multidisciplinary mass optimization [60]. The objective is to minimize a vehicle's mass, which incorporates decisions about materials, gauges, and vehicle shape with 68 performance constraints. The best-known optimum mass is approximately 222.74. We followed the implementation from Papenmeier et al. [61].[2]

B.5 LassoBench

LassoBench [62] [3] is a high-dimensional benchmark for hyperparameter optimization, specifically designed to tune the hyperparameters of the Weighted LASSO (Least Absolute Shrinkage and Selection Operator) regression model. It includes both synthetic tasks (simple, medium, high, and hard) and real-world tasks (Breast cancer, Diabetes, Leukemia, DNA, and RCV1). In this work, we focus on the DNA task, a microbiology classification problem. It computes the average validation error across all cross-validation folds as an unconstrained objective. We reformulate the problem by retaining the mean validation error as the objective while introducing constraints that the validation error on each fold must not exceed 0.32.

C Baselines Details

In this section, we provide a thorough description of our baseline implementation details and specify the hyperparameter settings used across all experiments.

DiffOPT [12]: As there is no publicly available code, we re-implement this baseline on our own. To approximate the data distribution, we use diffusion models with a similar architecture to our method. To enable accurate sampling from the target distribution, we implement Langevin dynamics as the energy function, which can be constructed by surrogate models in our setting, is differentiable.

SCBO [5]: We follow the tutorial code for SCBO provided by `botorch` [4] to reproduce the results.

PCAGP-SCBO [4]: To reproduce PCAGP-SCBO, we follow the code for SCBO and then apply `torch pca`[5] to project high-dimensional data into a reduced latent space with dimension l before fitting GP surrogates for constraints. For all synthetic tasks, we use $l = 2$ and for real-world tasks, we conduct a hyperparameter search on $[2, \lfloor D/2 \rfloor]$ and report the best one.

FuRBO [24]: We use the original implementation[6] for reproducing this baseline. FuRBO employs a feasibility-driven trust region strategy that adapts the search region based on constraint satisfaction.

cEI [21]: We implement cEI acquisition function by using `qExpectedImprovement()` in `botorch` library. We train a GP surrogate model independently for the objective and each constraint.

LogcEI [22]: We implement logcEI acquisition function by using `qLogExpectedImprovement()` in `botorch` library. We train a GP surrogate model independently for the objective and each constraint.

CMA-ES [25]: We employ the `pycma`[7] library [63]. For constraint handling, we formulate the problem using the same Lagrangian approach with the same λ value as ours for each task.

COBYLA [27]: We use the `scipy.optimize.minimize` implementation with `method='COBYLA'`. Constraints are passed directly to the optimizer in the form $-g^{(m)}(\mathbf{x}) \geq 0$ for each $m = 1, \ldots, M$.

Sobol [50]: We use Sobol quasi-random sequences generated via `SobolEngine()` as a non-adaptive space-filling baseline. At each step, the next point in the Sobol sequence is evaluated without any model-guided search.

[2] `https://github.com/LeoIV/BAxUS`
[3] `https://github.com/ksehic/LassoBench`
[4] `https://botorch.org/docs/tutorials/scalable_constrained_bo/`
[5] `https://github.com/valentingol/torch_pca`
[6] `https://github.com/paoloascia/FuRBO`
[7] `https://github.com/CMA-ES/pycma`

D Implementation Details

In this section, we introduce the implementation details of our method **CiBO**. Specifically, model architectures, the training processes employed, the hyperparameter configurations used, and the computational resources required.

D.1 Training Models

D.1.1 Training Proxies

We employ an ensemble of five proxies to model the objective function and a single proxy for each constraint. Each proxy is implemented as a MLP with three hidden layers of 1024 units, using GELU [64] activations. Proxies are trained with the Adam optimizer [65] for 100 epochs per round at a learning rate of 1×10^{-3} and a batch size of 256. All hyperparameters related to the proxy are listed in Table 1.

Table 1: Hyperparameters for Training Proxy

	Parameters	Values
Architecture	Num Ensembles	5
	Number of Layers	3
	Num Units	1024
Training	Batch size	256
	Optimizer	Adam
	Learning Rate	1×10^{-3}
	Training Epochs	100

D.1.2 Training Flow-based Models

We adopt the architecture of Lipman et al. [66] for our flow model, comprising three hidden layers with 512 units each. Training is performed using Adam optimizer for 500 epochs per round, with a learning rate of 1×10^{-3} and a batch size of 256. For ODE integration during sampling, we employ the Runge-Kutta 4 method with `torchdiffeq` [67], and set the integration steps as 250. All flow-model hyperparameters are detailed in Table 2.

Table 2: Hyperparameters for Training Flow-based Model

	Parameters	Values
Architecture	Number of Layers	3
	Num Units	512
Training	Batch size	256
	Optimizer	Adam
	Learning Rate	1×10^{-3}
	Training Epochs	500

D.2 Sampling Candidates

Training Diffusion Sampler Various approaches have been developed to draw samples from a distribution when only an unnormalized probability density or energy function is available. Traditional methods include Markov Chain Monte Carlo (MCMC) techniques [68, 69, 70, 71, 72, 73], though their computational cost increases dramatically in high-dimensional spaces. More recently, amortized variational inference methods, those based on training diffusion samplers [74, 75, 76, 77, 78, 79, 80], have gained widespread adoption as they offer improved scalability for high-dimensional problems.

Following the Venkatraman et al. [19], we adopt method from Sendera et al. [81] to train diffusion sampler to sample from the target:

$$p_{\text{post}}(\mathbf{z}) \propto p(\mathbf{z}) \exp\left(\beta \cdot r_\phi(f_\theta(\mathbf{z}))\right) \tag{15}$$

Here, the right-hand-side term serves as an unnormalized probability density, which the diffusion sampler amortizes the sampling cost by approximating it.

Off-policy Training of Diffusion Sampler As mentioned in the Section 4.2, we use the Trajectory Balance objective to train the diffusion sampler.

$$\mathcal{L}_{\text{TB}}(\mathbf{z}_{0:1}; \psi) = \left(\log \frac{Z_\psi p(\mathbf{z}_0) \prod_{i=0}^{T-1} p_F(\mathbf{z}_{(i+1)\Delta t} | \mathbf{z}_{i\Delta t}; \psi)}{p(\mathbf{z}_1) r(f_\theta(\mathbf{z}_1)) \prod_{i=1}^{T} p_B(\mathbf{z}_{(i-1)\Delta t} | \mathbf{z}_{i\Delta t})} \right)^2$$

The primary advantage of the TB loss is off-policy training [81, 82]. We can train our model not only from the on-policy trajectories through the reverse SDE $\{\mathbf{z}_0, \cdots, \mathbf{z}_1\} = \tau \sim p_F(\tau)$ but also from the trajectories through the forward SDE conditioned on the generated samples $\tau \sim p_B(\tau|\mathbf{z}_1)$. This proves its effectiveness on mode coverage and credit assignment [81].

Specifically, we repeat two processes. First, we sample trajectories on-policy $\tau \sim p_F(\tau)$, train the model with Equation (14), and collect the samples $\mathbf{z}_1$ into the buffer. Second, from the collected samples $\mathbf{z}_1$, we generate off-policy trajectories through $\tau \sim p_B(\tau|\mathbf{z}_1)$, then train with the Equation (14). During the off-policy training, we prioritize the samples with low energy: $\mathcal{E}(\mathbf{z}_1) = -\log(p(\mathbf{z}_1) r(f_\theta(\mathbf{z}_1)))$ following Sendera et al. [81] to make our model focus on the low energy samples. These techniques improve the overall performance of our framework (Appendix E.3).

We use the original code[8] released from Sendera et al. [81] for implementation. We also set method-specific hyperparameters with Path Integral Sampler (PIS) [74] architecture, zero initialization, and t-scale to 1 to make sure the initialized $p_F(\mathbf{z}_1)$ starts from the standard normal distribution. Detailed hyperparameters for training the diffusion sampler can be found in Table 3.

Table 3: Hyperparameters for Training Diffusion Sampler

	Parameters	Values
Architecture	Number of Layers	2
	Num Units	256
	Diffusion Time Steps	50
Training	Batch size	256
	Optimizer	Adam
	Learning Rate	1×10^{-3}
	Training Epochs	50

[8] https://github.com/GFNOrg/gfn-diffusion

D.3 Hyperparameters

In our formulation of constrained black-box problems, we introduce λ in Equation (11) for Lagrangian augmentation. We set $\gamma = 1$, which is the exploration bonus term that controls the Exploration-Exploitation trade-off. Inverse temperature β in Equation (12) balances the influence between prior and likelihood, and is fixed to 10^5. We draw $N \times B$ samples from the posterior distribution, then select B samples during filtering, where we fix $N = 10$ for all tasks. After evaluation, we update the training set by keeping the top L highest-scoring samples subject to the Lagrangian objective. We set $L = 2000$ for synthetic tasks and $L = 1000$ for real-world tasks. The reweighting hyperparameter is set to $\kappa = 0.01$ for synthetic tasks and $\kappa = 0.1$ for real-world tasks. The Lagrange multiplier λ is crucial for balancing objectives and constraints, and we report the task-specific values in Table 4. We include additional analysis to assess how each parameter affects overall performance in Section 5.6 and Appendix E.

Table 4: Lagrange multiplier λ for each task.

	Lambda λ
Ackley 200D	10
Rastrigin 200D	10
Rosenbrock 200D	10
RoverPlanning 60D	5
HalfCheetah 102D	5
Mopta 124D	10
DNA 180D	5

E Further Analysis

In this section, we provide further analysis on different components of our method that are not included in the main manuscript due to the page limit.

E.1 Analysis on Feasibility Ratio

To further analyze our method's ability to effectively handle constraints, we report the feasibility ratio across optimization batches for the Rastrigin 200D and RoverPlanning 60D tasks.

As illustrated in Figure 7, CiBO achieves the highest feasibility ratio within 5 to 10 batches, significantly faster than all competitors. While some baselines such as SCBO and PCAGP-SCBO eventually reach comparable ratios, they require about twice as many batches. In the Rover Planning tasks, most baselines fail to exceed a feasibility ratio of 0.4, whereas CiBO shows steady improvement and maintains high feasibility throughout the optimization process.

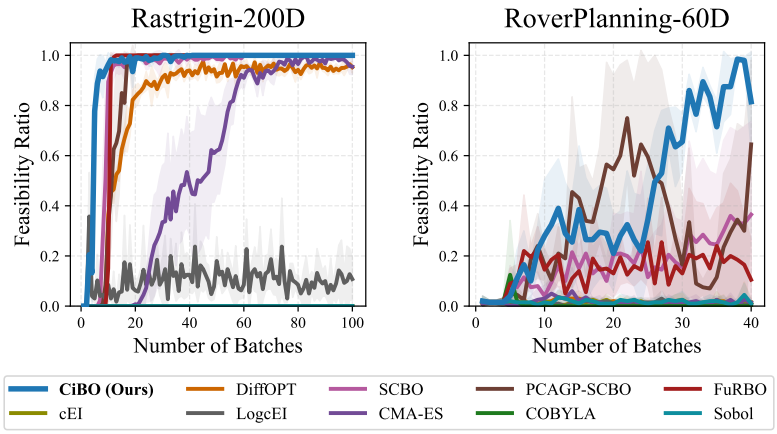


Figure 7: Feasibility ratio over all baselines. Experiments are conducted with four random seeds, and the mean and one standard deviation are reported.

E.2 Analysis on Reweighting Strategy

We compare different reweighting strategies for training the generative model: score-weighted sampling that weights each sample proportionally to its Lagrangian-augmented value $w(y, c, D_r)$, and rank-based reweighting (ours) as described in Equation (7). As shown in Figure 8, rank-based reweighting outperforms score-based alternatives across both synthetic and real-world tasks.

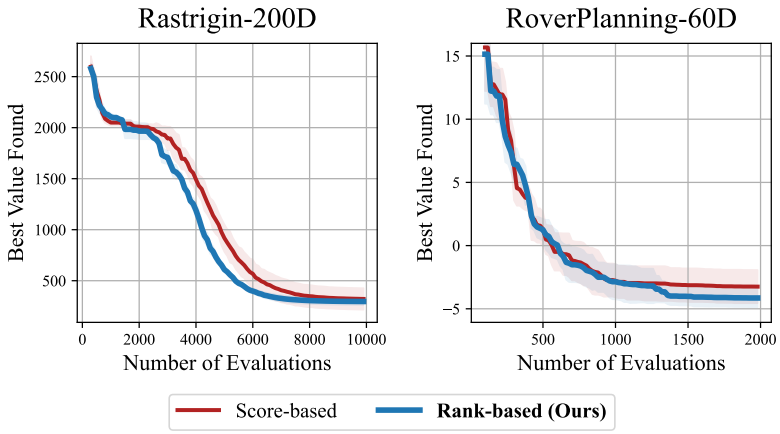


Figure 8: Comparison of reweighting strategies in Rastrigin-200D and Rover Planning-60D. Rank-based reweighting consistently outperforms score-based reweighting. Experiments are conducted with four random seeds, and the mean and one standard deviation are reported.

E.3 Effect of Off-policy Training in Amortized Inference

We employ off-policy training with the TB loss to train the diffusion sampler. To analyze the impact of off-policy training on performance, we conduct ablation studies on different training schemes. As shown in Figure 9, off-policy training consistently outperforms on-policy methods, and the performance gap widens as the number of constraints grows (15 constraints in Rover Planning versus only 2 in Rastrigin). It highlights that training with off-policy samples is crucial for amortizing the posterior distribution, which has multiple modes and a large plateau, by improving mode coverage.

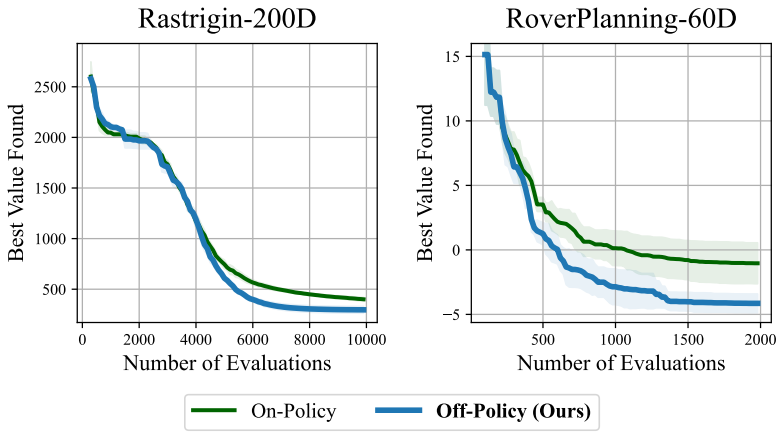


Figure 9: Comparison between off-policy and on-policy in Rastrigin-200D and Rover Planning-60D. Experiments are conducted with four random seeds, and the mean and one standard deviation are reported.

E.4 Analysis on Adaptive λ

A natural question is whether adaptively updating the Lagrange multiplier λ during optimization can improve performance, as is common in augmented Lagrangian methods. To investigate this, we compare our fixed λ against an adaptive variant that updates λ based on the observed constraint violations at each round. Specifically, we increase λ when the feasibility ratio of the current batch is low and decrease it otherwise. As shown in Figure 10, the fixed λ performs comparably to or better than the adaptive variant on both Rastrigin-200D and RoverPlanning-60D. On RoverPlanning, the fixed λ achieves a notably better final value. We attribute this to the fact that adaptive updates introduce additional sensitivity to the noisy feasibility estimates in early rounds, whereas a fixed λ provides a stable training signal throughout optimization.

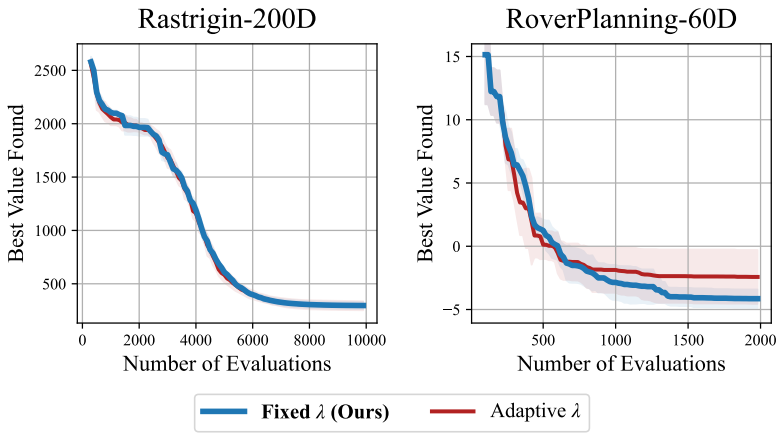


Figure 10: Comparison between fixed and adaptive λ on Rastrigin-200D and RoverPlanning-60D. Experiments are conducted with four random seeds, and the mean and one standard deviation are reported.

E.5 Analysis on Buffer Size L

In each round, we retain the L top-scoring samples with respect to the Lagrangian-relaxed objective function for computational efficiency. To analyze the effect of the buffer size L, we conduct experiments by varying L. As demonstrated in Figure 11, using too small L occasionally gets stuck in a sub-optimal solution, while using too large L exhibits a slow convergence rate. Notably, the final performance of CiBO is not significantly influenced by the buffer size, indicating a low sensitivity to the choice of L.

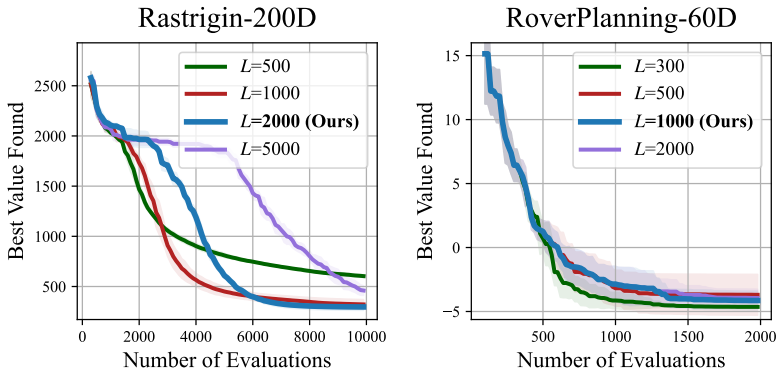


Figure 11: Performance of CiBO in Rastrigin-200D and Rover Planning-60D with varying L. Experiments are conducted with four random seeds, and the mean and one standard deviation are reported.

E.6 Analysis on Initial Dataset size $|\mathcal{D}_0|$ and Batch size B

The size of the initial dataset, $|\mathcal{D}_0|$, and the batch size, B, play a critical role in the performance of black-box optimization algorithms. When $|\mathcal{D}_0|$ is small and B is large, the algorithm must optimize using very limited information, making the search significantly more challenging. To this end, we conduct experiments varying $|\mathcal{D}_0|$ and B to demonstrate the robustness of our method on initial data configurations. As shown in Figure 12, our method demonstrates robustness regarding both the initial dataset size $|\mathcal{D}_0|$ and the batch size B.

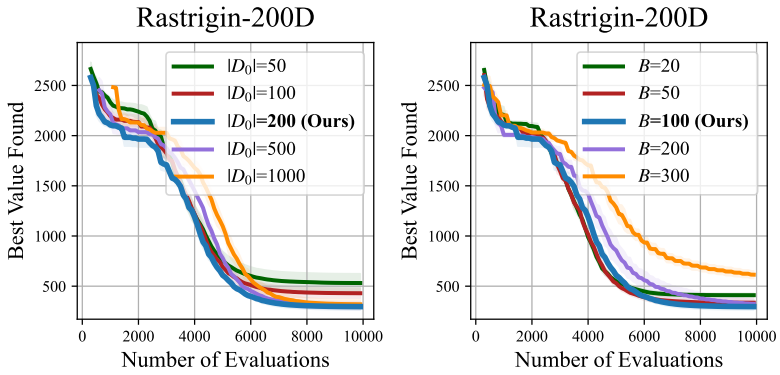


Figure 12: Performance of CiBO in Rastrigin-200D with varying $|D_0|$ and B. Experiments are conducted with four random seeds, and the mean and one standard deviation are reported.

E.7 Analysis on Misaligned Constraints

In the synthetic experiments, the optimal points of Rastrigin and Ackley are located at $\mathbf{x} = [0, 0, ..., 0]$. Therefore, for those functions, directly optimizing the second constraint in Appendix B.1 also optimizes the objective function. To verify that our method's performance does not simply benefit from this alignment between constraints and objectives, we conduct additional analysis on both the Rastrigin and Ackley functions with misaligned constraints: $\sum_{d=1}^{200}(x_d - 1) \leq 0$ and $||\mathbf{x} - [1, 1, ..., 1]||_2^2 \leq 30$, where the constraint centers are shifted away from the optimal point.

As shown in Figure 13, our method consistently outperforms the baselines on both tasks even with this misaligned constraint configuration. This demonstrates that the strong performance of our method stems from its expressive capacity to handle complex constraint landscapes, rather than benefiting from any particular alignment between constraint centers and objective optima.

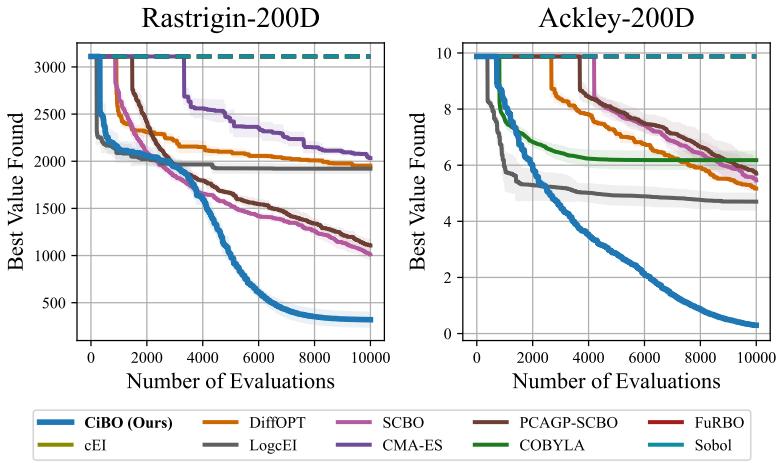


Figure 13: Analysis on misaligned constraints for Rastrigin-200D (left) and Ackley-200D (right). Experiments used four random seeds, reporting mean and one standard deviation.

E.8 Comparison with Unconstrained Generative Methods

DDOM [30] and DiBO [15] are generative model-based methods originally designed for unconstrained black-box optimization. To enable a comparison in our constrained setting, we adapt both methods by incorporating Lagrangian relaxation, searching $\lambda \in \{1, 2, 5, 10, 20\}$ and reporting results with the best λ per task. As shown in Figure 14, even with the best λ selected per task, DiBO and DDOM both underperform CiBO. These results demonstrate that simply augmenting unconstrained methods with Lagrangian relaxation is insufficient for constrained settings, underscoring the need for constraint-aware optimization framework.

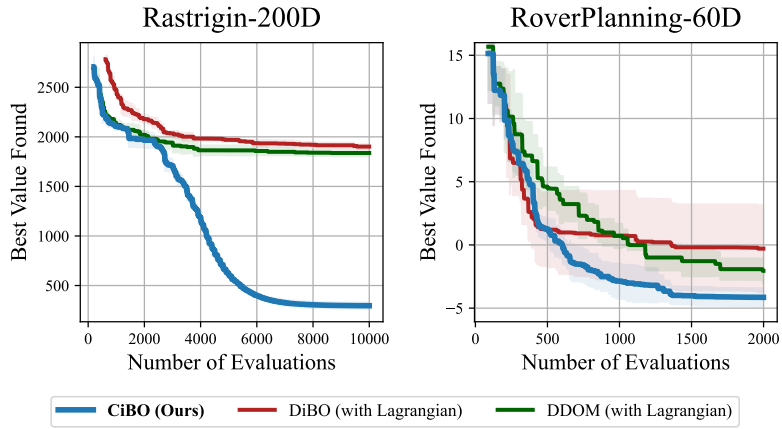


Figure 14: Comparison with unconstrained generative methods. Experiments are conducted with four random seeds, and the mean and one standard deviation are reported.

F Runtimes

We report the per-round wall-clock time of each method in Tables 5 and 6. We decompose each optimization round into four stages. *Surrogate* measures the time spent fitting a surrogate model to the observed data (e.g., GP fitting for BO-based methods, proxy ensemble training for generative methods). *Generative* captures the time for training a generative model that learns to propose candidates (applicable only to CiBO and DiffOPT). *Candidates* is the time for proposing a batch of candidate solutions from the trained models or search procedure. *Oracle* records the wall-clock time of evaluating the proposed candidates on the black-box objective and constraints. All experiments run on a single NVIDIA RTX 3090 GPU and Intel Xeon Platinum CPU @ 2.90 GHz, pinned to 8 cores for fair comparison.

As shown in the tables, CiBO requires mostly less time per round than standard BO-based methods, because the lightweight proxy ensemble avoids the expensive GP fitting that dominates the runtime of those approaches.

Table 5: Per-round wall-clock time (seconds) on Rastrigin-200D. We report the mean $\pm$ one standard deviation over 4 seeds.

	Surrogate	Generative	Candidates	Oracle	Total
cEI	128.15 ± 10.58	—	11.82 ± 0.16	<0.01	140.05 ± 10.56
LogcEI	127.82 ± 10.06	—	52.25 ± 7.28	<0.01	180.12 ± 6.08
SCBO	127.95 ± 10.06	—	3.70 ± 0.00	<0.01	131.65 ± 10.06
PCAGP-SCBO	129.07 ± 10.79	—	3.70 ± 0.00	<0.01	132.77 ± 10.79
FuRBO	127.85 ± 10.09	—	4.43 ± 0.04	<0.01	132.35 ± 10.09
CMA-ES	—	—	0.10 ± 0.00	<0.01	0.10 ± 0.00
COBYLA	—	—	0.10 ± 0.00	<0.01	0.10 ± 0.00
Sobol	—	—	0.10 ± 0.00	<0.01	0.10 ± 0.00
DiffOPT	53.47 ± 0.78	12.05 ± 0.05	0.60 ± 0.00	<0.01	66.12 ± 0.83
CiBO	$\mathbf{25.60 \pm 0.17}$	$\mathbf{20.27 \pm 0.26}$	$\mathbf{0.90 \pm 0.00}$	<0.01	$\mathbf{46.83 \pm 0.36}$

Table 6: Per-round wall-clock time (seconds) on RoverPlanning-60D. We report the mean $\pm$ one standard deviation over 4 seeds.

	Surrogate	Generative	Candidates	Oracle	Total
cEI	38.38 ± 2.58	—	25.32 ± 0.31	0.60 ± 0.00	64.35 ± 2.92
LogcEI	38.50 ± 2.62	—	34.25 ± 14.12	0.62 ± 0.04	73.38 ± 15.01
SCBO	38.75 ± 2.72	—	7.75 ± 0.05	0.60 ± 0.00	47.15 ± 2.72
PCAGP-SCBO	10.25 ± 1.82	—	1.80 ± 0.00	0.60 ± 0.00	12.70 ± 1.77
FuRBO	39.88 ± 2.73	—	8.00 ± 0.00	0.60 ± 0.00	48.55 ± 2.77
CMA-ES	—	—	<0.01	0.60 ± 0.00	0.60 ± 0.00
COBYLA	—	—	0.10 ± 0.00	0.60 ± 0.00	0.70 ± 0.00
Sobol	—	—	0.10 ± 0.00	0.60 ± 0.00	0.70 ± 0.00
DiffOPT	33.30 ± 2.92	2.40 ± 0.00	1.10 ± 0.00	0.62 ± 0.04	37.40 ± 2.92
CiBO	$\mathbf{16.43 \pm 0.40}$	$\mathbf{16.35 \pm 0.15}$	$\mathbf{0.90 \pm 0.00}$	0.60 ± 0.00	$\mathbf{34.30 \pm 0.46}$

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the main contributions: reformulating constrained black-box optimization as posterior inference in the latent space of flow-based models, and amortizing sampling via an outsourced diffusion sampler. These claims are supported by extensive experiments on synthetic and real-world benchmarks in Section 5.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations and future work are discussed at the end of the Conclusion (Section 6). We acknowledge the need to retrain all models each round and identify potential improvements from recent advances in flow-based model training and diffusion samplers.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: This paper does not include new theoretical results. Our method builds on established results from outsourced diffusion sampling [19] and the trajectory balance objective [20], which are properly cited.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide full implementation details in Appendix D, including model architectures, hyperparameters, and training procedures. Task descriptions are in Appendix B, and baseline configurations are in Appendix C. We also release our code via an anonymous repository.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We provide anonymous access to our code repository at `https://anonymous.4open.science/r/CiBO-3869/`, which includes the implementation of our method and scripts to reproduce the experiments.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: All experimental settings are described in Section 5 and the appendix. Task details are in Appendix B, baseline configurations in Appendix C, and full implementation details (architectures, optimizers, hyperparameters) in Appendix D.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: All experiments are conducted with multiple random seeds (4 seeds for real-world tasks, 12 seeds for synthetic tasks). We report the mean and one standard deviation across seeds in all figures and tables.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix F reports the per-round wall-clock time for all methods. All experiments are run on a single NVIDIA RTX 3090 GPU and Intel Xeon Platinum CPU @ 2.90 GHz with 8 cores, as stated in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and confirm that our research conforms to it in every respect.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [N/A]

Justification: This work presents a general-purpose optimization framework. It is foundational research not tied to a specific application, and we do not foresee direct negative societal impacts beyond those inherent to any optimization method.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: Our work does not involve releasing pre-trained language models, image generators, or scraped datasets. The released code is for a general-purpose optimization algorithm and does not pose a high risk for misuse.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All benchmark tasks and baseline methods are properly cited. The benchmarks used (RoverPlanning, HalfCheetah, Mopta, LassoBench) are cited with their original publications, and LassoBench is referenced with its repository URL.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We release our code via an anonymous repository with instructions for reproducing the experiments. The repository includes implementation details and configuration files.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: This paper does not involve crowdsourcing or research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: This paper does not involve crowdsourcing or research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are not used as a component of the core methodology in this research.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.