DIFF-BBO: DIFFUSION-BASED INVERSE MODELING FOR BLACK-BOX OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Black-box optimization (BBO) aims to optimize an objective function by iteratively querying a black-box oracle in a sample-efficient way. While prior studies focus on forward approaches to learn surrogates for the unknown objective function, they struggle with steering clear of out-of-distribution and invalid inputs. Recently, inverse modeling approaches that map objective space to the design space with conditional diffusion models have demonstrated impressive capability in learning the data manifold. They have shown promising performance in offline BBO tasks. However, these approaches require a pre-collected dataset. How to design the acquisition function for inverse modeling to *actively* query new data remains an open question. In this work, we propose *diffusion-based inverse modeling for black-box optimization* (Diff-BBO), an *inverse* approach leveraging diffusion models for online BBO problem. Instead of proposing candidates in the design space, Diff-BBO employs a novel acquisition function *Uncertainty-aware Exploration* (UaE) to propose objective function values. Subsequently, we employ a conditional diffusion model to generate samples based on these proposed values within the design space. We demonstrate that using UaE results in optimal optimization outcomes, supported by both theoretical and empirical evidence.

1 INTRODUCTION

Practical problems in science and engineering often involve optimizing a black-box objective function that is expensive to evaluate, seen in fields such as neural network architecture design (Zoph & Le, 2016), robotics (Tesch et al., 2013), and molecular design (Sanchez-Lengeling & Aspuru-Guzik, 2018). How to achieve a near-optimal solution while minimizing function evaluations is thus a major challenge in black-box optimization (BBO). To improve sample efficiency, prior works in BBO have largely focused on the online setting where the algorithm can iteratively select candidates in the design space and query the black-box function for evaluation (Turner et al., 2021; Zhang et al., 2021; Hebbal et al., 2019; Mockus, 1974). Most existing algorithms belong to the class of forward methods, including Bayesian optimization (BO) (Kushner, 1964; Mockus, 1974; Wu et al., 2023; Frazier, 2018), bandit algorithms (Agrawal & Goyal, 2012; Karbasi et al., 2023), and conditional sampling approaches (Brookes et al., 2019; Gruver et al., 2024; Stanton et al., 2022). They build a surrogate model to approximate and optimize the black-box function sequentially.

However, these approaches may face difficulties in scenarios where valid inputs represent a small subspace, such as valid protein sequences or molecular structures. Such optimization problems become exceptionally challenging, as the optimizer must navigate and avoid out-of-distribution and invalid inputs (Kumar & Levine, 2020). Recently, a novel set of methods, termed *inverse approaches*, have been proposed to address this issue. These methods (Kumar & Levine, 2020; Krishnamoorthy et al., 2023; Kim et al., 2023) break the traditional paradigm by learning an inverse mapping from the objective space back to the input space (a.k.a., the black-box function's design space). Leveraging state-of-the-art generative models, such as diffusion models (Song et al., 2020), these approaches effectively capture data distributions in the design space and facilitate optimization within the data manifold (Kong et al., 2024). Besides, diffusion models naturally provide uncertainty estimates through the probabilistic nature of the diffusion process (Chan et al., 2024; Du & Li, 2023), which can be further utilized to design informative exploration strategies to propose better candidate solutions for optimization problems. They achieve high performance in *offline* optimization settings (Kumar & Levine, 2020; Lu et al., 2023; Wang et al., 2018), assuming access to a fixed pre-collected dataset.

Note that the offline setting can be restrictive compared with the *online* setting, which allows for continuous learning and improvements from new data samples.

Despite the advancements in the offline setting, we cannot directly apply inverse modeling approaches to the online setting due to the unresolved issues regarding how to accurately capture the uncertainty of the inverse model and design an acquisition function for data-efficient querying. In this paper, we propose Diff-BBO, an inverse approach for online black-box optimization (BBO). Diff-BBO places a distribution within the design space and represents it with a conditional diffusion model. Although diffusion model necessitates a relatively large dataset to effectively learn the data manifold, we show that the low-quality pre-collected dataset with average or below-average objective function values suffices for the initial training stage of Diff-BBO. Our approach consists of a novel acquisition function *Uncertainty-aware Exploration* (UaE), which leverages the uncertainty of the conditional diffusion model to strategically propose the desired objective function values for sampling the design space. We summarize our main contributions as follows:

- We present Diff-BBO, an inverse modeling approach for efficient online black-box optimization (BBO) leveraging uncertainty of conditional diffusion models.
- We provide an uncertainty decomposition into epistemic uncertainty and aleatoric uncertainty for conditional diffusion models. We rigorously analyze how uncertainty propagates throughout the denoising process of conditional diffusion model.
- We design a novel acquisition function UaE for Diff-BBO. Theoretically, we prove that the balance between targeting higher objective values and minimizing the epistemic uncertainty lead to optimal optimization outcomes.
- We demonstrate that Diff-BBO achieves state-of-the-art performance with superior sample efficiency on Design-Bench and molecular discovery task in the online BBO setting.

2 Related work

Black-box Optimization. While recent studies aim to solve offline Black-box Optimization (BBO) using a pre-collected dataset (Li et al., 2024; Krishnamoorthy et al., 2023; Fu & Levine, 2021) without querying the oracle function, prior works in BBO have largely focused on the online setting where a model can iteratively query the function during training (Turner et al., 2021; Zhang et al., 2021; Hebbal et al., 2019; Mockus, 1974). In both settings, most existing algorithms belong to the class of forward methods, including Bayesian optimization (BO) (Kushner, 1964; Mockus, 1974; Wu et al., 2023; Frazier, 2018), bandit algorithms (Agrawal & Goyal, 2012; Karbasi et al., 2023), and conditional sampling approaches (Brookes et al., 2019; Gruver et al., 2024; Stanton et al., 2022). Liu et al. (2024) further integrate LLM capabilities into the BO framework to enable zero-shot warmstarting and enhance surrogate modeling and candidate sampling, thereby improving the sample efficiency. Forward methods build a surrogate model to approximate and optimize the black-box objective function. However, these approaches can struggle with capturing the data manifold within the design space and avoiding out-of-distribution and invalid inputs (Kumar & Levine, 2020). Song et al. (2022); Zhang et al. (2021) proposed Likelihood-free BO using likelihood-free inference to extend BO to a broader class of models and utilities. It directly models the acquisition function without separately performing inference with a surrogate model. However, there is a risk where the acquisition function is over-confident. Our work builds upon recent progress in inverse approaches for offline BBO, which utilize diffusion modeling to better learn the data manifold within the design space (Krishnamoorthy et al., 2023; Kong et al., 2024). But we focus solely on online BBO setting by introducing a sample-efficient inverse modeling method using conditional diffusion models.

Diffusion Models. As an emerging class of generative models with strong expressiveness, diffusion models (Sohl-Dickstein et al., 2015; Song et al., 2020) have been successfully deployed across various domains including image generation (Rombach et al., 2022), reinforcement learning (Wang et al., 2022), robotics (Chi et al., 2023), etc. Notably, through the formulation of stochastic differential equations (SDEs), (Song et al., 2020) provides a unified continuous-time score-based framework for distinctive classes of diffusion models. To steer the generation toward high-quality samples with desired properties, it is important to guide the backward data-generation process using task-specific information. Hence, different types of guidance are studied in prior works (Bansal et al., 2023; Nichol et al., 2021; Zhang et al., 2023), including classifier guidance (Dhariwal & Nichol, 2021) where the classifier is trained externally, and classifier-free guidance (Ho & Salimans, 2022), in which the classifier is implicitly specified. In this work, we employ classifier-free guidance to eliminate the requirement of training a separate classifier model, thereby enabling feasible uncertainty quantification in conditional diffusion models.

Uncertainty Quantification. Uncertainty quantification (UQ) often relies on probabilistic modeling, with Bayesian approximation and ensemble learning being two popular types of approaches. Bayesian Neural Networks (BNNs) (MacKay, 1992; Neal, 2012; Kendall & Gal, 2017; Zhang et al., 2018) employ variational inference (VI) to sample model weights from a tractable distribution and estimate uncertainty through sample variance. When training large-scale models, Monte Carlo (MC) dropout (Srivastava et al., 2014) offers a cost-effective alternative by approximating BNNs during inference (Gal & Ghahramani, 2016). On the other hand, deep ensembles (Lakshminarayanan et al., 2017) train multiple NNs with different initial weights to gauge uncertainty via model variance, which also faces scalability issues as network size increases. To address this issue, recent efforts incorporate ensembling techniques in generative models to separate uncertainty into aleatoric and epistemic components (Valdenegro-Toro & Mori, 2022; Ekmekci & Cetin, 2023). To further improve the scalability of deep ensembles, (Chan et al., 2024) proposed hyper-diffusion to quantify the uncertainty with a single diffusion model. In comparison, we take one step further by utilizing the quantified uncertainty of conditional diffusion models to solve the black-box optimization problem as a downstream task.

3 Preliminaries

3.1 Problem Formulation

Let $f : \mathcal{X} \to \mathbb{R}$ denote the unknown ground-truth black-box function that evaluates the quality of any data point $\boldsymbol{x}$, with $\mathcal{X} \subseteq \mathbb{R}^d$. Our goal is to find the optimal point $\boldsymbol{x}^*$ that maximizes f:

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

We are interested in the online BBO setting in which f is expensive to evaluate and the number of evaluations is limited. In particular, we consider batch online BBO. With a fixed query budget of K and batch size N, we iteratively query f with N new inputs in each batch, and update the surrogate model of f based on observed outputs within K iterations. A key concept in online BBO is the acquisition function, which guides the selection of new query points by balancing exploration and exploitation. This function aims to efficiently identify high-performing inputs, thereby efficiently solving the online optimization problem.

3.2 Conditional Diffusion Model

Diffusion Models (Sohl-Dickstein et al., 2015; Song et al., 2020) are probabilistic generative models that learn distributions through an iterative denoising process. These models consist of three components: a forward diffusion process that produces a series of noisy samples by adding Gaussian noise, a reverse process to reconstruct the original data samples from the noise, and a sampling procedure to generate new data samples from the learned distribution. Let the original sample be $\mathbf{x}_0$ and t be the diffusion step. For conditional diffusion models, a conditional variable y is added to both the forward process as $q\left(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, y\right)$ and reverse process as $p_\theta\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, y\right),\ \forall t \in [T]$.

The reverse process begins with the standard Gaussian distribution $p(\boldsymbol{x}_T) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, and denoises $\boldsymbol{x}_t$ to recover $\boldsymbol{x}_0$ through the following Markov chain with reverse transitions:

$$p_\theta\left(\boldsymbol{x}_{0:T}|y\right) = p(\boldsymbol{x}_T)\prod_{t=1}^{T} p_\theta\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, y\right), \quad \boldsymbol{x}_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}),$$
$$p_\theta\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, y\right) = \mathcal{N}\left(\boldsymbol{x}_{t-1}; \mu_\theta(\boldsymbol{x}_t, t, y), \Sigma_\theta(\boldsymbol{x}_t, t, y)\right).$$

During training, Σ_θ is empirically fixed, and μ_θ is reparametrized by a trainable denoise function $\boldsymbol{\epsilon}_\theta\left(\boldsymbol{x}_t, t, y\right)$, which is used to estimate the noise vector ϵ that was added to input $\boldsymbol{x}_t$, and is trained by minimizing a reweighted version of the evidence lower bound (ELBO):

$$\mathcal{L}_{\text{dif}} = \mathbb{E}_{\boldsymbol{x}_0\sim q(\boldsymbol{x}), y, \boldsymbol{\epsilon}\sim\mathcal{N}(0,\boldsymbol{I}), t\sim\mathcal{U}(0,T), \boldsymbol{x}_t\sim q(\boldsymbol{x}_t|\boldsymbol{x}_0,y)}\left[w\left(t\right)\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta\left(\boldsymbol{x}_t, t, y\right)\right\|_2^2\right]. \tag{2}$$

Note that the loss in Equation (2) (Ho et al., 2020) for $\boldsymbol{\epsilon}_\theta$ is denoising score matching for all time step t, which estimates the gradient of the log probability density of the noisy data (a.k.a. score function): $\boldsymbol{\epsilon}_\theta\left(\boldsymbol{x}_t, t, y\right) \approx -\sigma_t \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x} \mid y)$. We further denote the score function as $s_\theta(\boldsymbol{x}_t, y, t) := -\boldsymbol{\epsilon}_\theta\left(\boldsymbol{x}_t, t, y\right)/\sigma_t$.

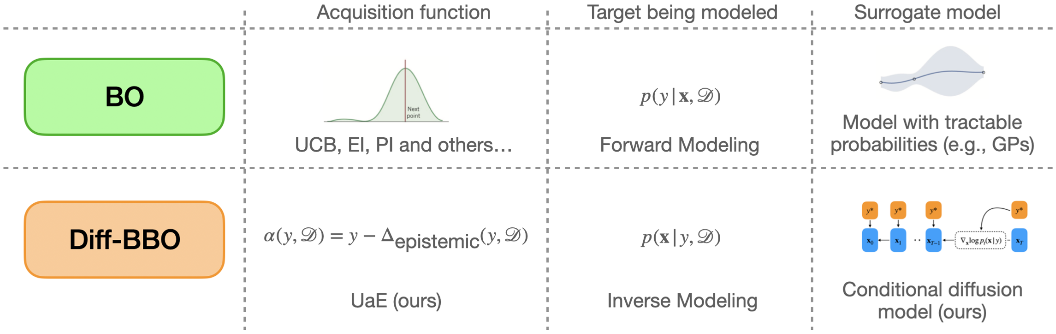

Figure 1: Forward modeling vs inverse modeling for black-box optimization. (*Top*) Forward modeling approach using surrogate models (e.g., GPs) for forward modeling and acquisition functions (e.g., UCB, PI, and EI) to select $\mathbf{x}$. (*Bottom*) Our inverse modeling approach using generative model (e.g., diffusion model) for inverse modeling and acquisition function (e.g., UaE) to select y.

Algorithm 1: Diff-BBO

Input: Initial dataset $\mathcal{D} = \{\mathbf{x}, y\}$, total number of iterations K, candidate feasible range C, oracle function $f(\cdot)$, batch size N

Initialization: Conditional diffusion model $p_\theta(\mathbf{x}|y)$

for $k = 1, 2, \cdots K$ **do**

- Train the conditional diffusion model with $\mathcal{D}$
- Construct a candidate set $\mathcal{Y} = \{y : 0 \leq y \leq C\}$
- $y_k^* = \text{argmax}_{y \in \mathcal{Y}}\, \alpha(y, \mathcal{D})$
- Generate $\{\mathbf{x}_j\}_{j=1}^N$ where $\mathbf{x}_j \sim p_\theta(\mathbf{x} \mid y_k^*, \mathcal{D})$
- Query the oracle function $f(\cdot)$ with generated samples $\{\mathbf{x}_j\}_{j=1}^N$
- $\mathcal{D} \leftarrow \mathcal{D} \cup \{\mathbf{x}_j, f(\mathbf{x}_j)\}_{j=1}^N$
- $\phi_k \leftarrow \max\left(f(x)\right)\ \ s.t.\ \ x \in \mathcal{D}$

Output: Reconstructed $\{\phi_k\}_{k=1}^K$

4 Diff-BBO

In this section, we present *Diffusion-based Inverse Modeling for Black-Box Optimization* (Diff-BBO), followed by the details of training its conditional diffusion model in the Bayesian setting.

4.1 The Diff-BBO Framework

A key distinction of Diff-BBO is to solve the online BBO problem in the inverse modeling setting, whereas prior works mainly focus on the forward modeling setting. In the latter, a surrogate model $p(y|\boldsymbol{x}, \mathcal{D})$ for the unknown objective f is learnt by utilizing models such as GPs. These methods typically rely on heuristic approaches to generate new candidate solutions, which can lead to out-of-distribution and invalid designs. In contrast, our approach leverages the power of diffusion models to represent $p(\boldsymbol{x}|y, \mathcal{D})$, allowing it to provide high-quality candidate solutions in the design space and to leverage arbitrary function values as conditional information. Besides, diffusion models naturally provide uncertainty estimates, which are further utilized in our design of the acquisition function. Figure 1 demonstrates a detailed comparison of Diff-BBO with prior forward methods to solve the online BBO problem.

Given this inverse modeling setting, we model the conditional distribution of $p(\boldsymbol{x}|y, \mathcal{D})$ with training data $\mathcal{D}$. The function value y to condition on is proposed by an acquisition function, which quantifies the quality of the generated $\boldsymbol{x}$. As the optimization performance at each iteration matters, the optimization objective given in Equation (1) becomes:

$$\max_{y_k \in \mathbb{R}} \sum_{k=1}^{K} f(\boldsymbol{x}_k), \quad \boldsymbol{x}_k \sim p_\theta(\cdot \mid y_k, \mathcal{D}), \quad \theta \in \Theta. \tag{3}$$

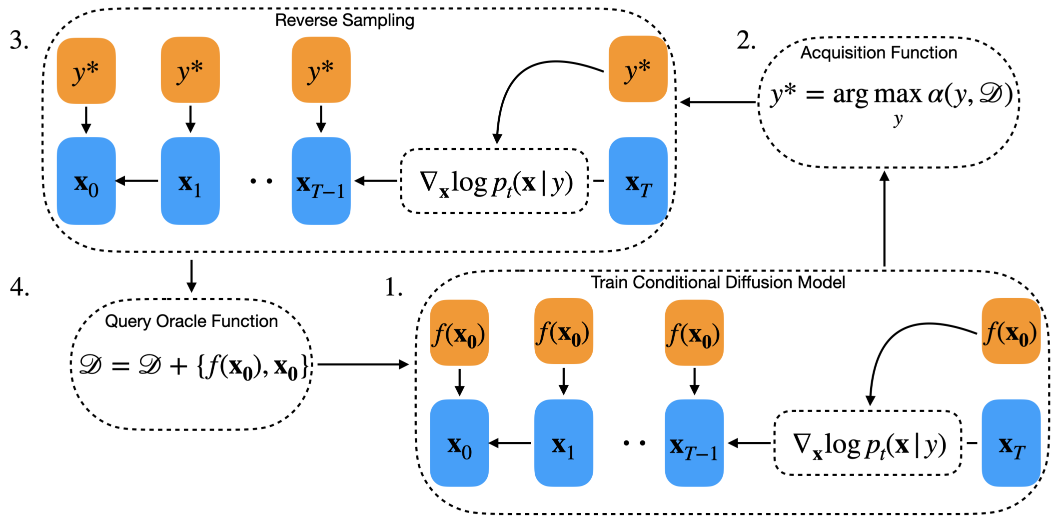

Figure 2: Black-box optimization framework using the conditional diffusion model as the inverse model. The overall framework includes 4 stages. 1. Train the conditional diffusion model given the current training dataset. 2. Compute the acquisition function and select the optimal y^* to condition on. 3. Generate samples $\{\mathbf{x_0}\}$ conditioned on y^*. 4. Query the oracle given generated samples $\{\mathbf{x_0}\}$ and update the training dataset.

To solve the above optimization problem, we introduce Diff-BBO in Algorithm 1. At each iteration k, we train a conditional diffusion model and compute the optimal y_k^* with the designed acquisition function. In practice, we select $y = w \cdot \phi_k$ from a constructed candidate set $\mathcal{Y}$ based on the acquisition function scores $\alpha(y)$, where the weight w belongs to a set of positive scalars $\mathcal{W}$ and ϕ_k is the maximum function values being queried in the current training dataset $\mathcal{D}$. Conditioning on y_k^*, we generate N samples $\{\mathbf{x}_j\}_{j=1}^N$, where $\boldsymbol{x}_j \sim p_\theta(\mathbf{x}|y_k^*, \mathcal{D})$. By querying the black-box oracle to evaluate each $\boldsymbol{x}_j$, we obtain the best possible reconstructed value ϕ_k for the current iteration, and append all queried data pairs $\{\mathbf{x}_j, f(\mathbf{x}_j)\}_{j=1}^N$ to the training dataset $\mathcal{D}$. The overall Diff-BBO framework with the conditional diffusion model is shown on Figure 2.

4.2 Conditional Diffusion Model Training in Bayesian Setting

Instead of estimating a set of deterministic parameters θ from a deterministic neural network, we are interested in learning its Bayesian posterior to further understand and improve the model's performance as well as its reliability with uncertainty quantification. In Bayesian settings, we consider the model parameters $\theta \in \Theta$, where Θ is the parameter space, and maintain its posterior distribution $p(\theta|\mathcal{D})$, which is learned from training data $\mathcal{D}$. By choosing θ from its posterior, essentially we sample a score function $\widetilde{s}_\theta(\boldsymbol{x}_t, y, t)$ from the probability distribution $p(s_\theta \mid \boldsymbol{x}_t, y, t, \mathcal{D}) = \mathcal{N}(s_\theta(\boldsymbol{x}_t, y, t), \Sigma_{s_\theta}(\boldsymbol{x}_t, y, t))$, whose expected value is $s_\theta(\boldsymbol{x}_t, y, t)$, and variance is a diagonal covariance matrix $\Sigma_{s_\theta}(\boldsymbol{x}_t, y, t)$.

Specifically, we adopt classifier-free guidance as in (Ho & Salimans, 2022) to eliminate the requirement of training a separate classifier model. We jointly train an unconditional diffusion model $p_\theta(\boldsymbol{x})$ parameterized by $\epsilon_\theta(\mathbf{x}, t, \emptyset)$ and a conditional diffusion model $p_\theta(\boldsymbol{x}|y)$ parameterized by $\epsilon_\theta(\mathbf{x}, t, y)$ by minimizing the following loss function:

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{x}_0, y, \boldsymbol{\epsilon}, t, \boldsymbol{x}_t, \lambda}\left[w\left(t\right)\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta\left(\boldsymbol{x}_t, t, (1-\lambda)y + \lambda\emptyset\right)\right\|_2^2\right], \quad (4)$$

where $\boldsymbol{x}_0 \sim q(\boldsymbol{x}), \boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I}), t \sim \mathcal{U}(0, T), \boldsymbol{x}_t \sim q(\boldsymbol{x}_t \mid \boldsymbol{x}_0), \lambda \sim \text{Bernoulli}(p_{\text{uncond}})$, and p_{uncond} is the probability of setting y to the unconditional information $\emptyset$.

5 Acquisition Function Design

In this section, we propose a novel acquisition function called *Uncertainty-aware Exploration* (UaE) for Diff-BBO. We first analyze the uncertainty of Diff-BBO from both theoretical and practical perspectives, decomposing the uncertainty into the aleatoric and epistemic components. Based on the uncertainty decomposition, we then propose UaE. We prove that by achieving a balance between high objective values and low epistemic uncertainty, UaE effectively provides a near-optimal solution to the online BBO problem.

5.1 Uncertainty Quantification on Conditional Diffusion Model

The optimization problem defined in Equation (3) presents a probabilistic formulation of the online BBO problem using inverse modeling. Instead of searching for a single optimal point $\boldsymbol{x}$, it aims to learn a parameterized distribution $p_\theta(\boldsymbol{x} \mid y, \mathcal{D})$ for a given y, and sampling from this predictive distribution. As such, we resort to the tools of Bayesian inference to solve this task. More specifically, given an observed value y of a sample $\boldsymbol{x}$, the objective of Bayesian inference is to estimate the predictive distribution:

$$p(\boldsymbol{x} \mid y, \mathcal{D}) = \mathbb{E}_\theta[p_\theta(\boldsymbol{x} \mid y)] = \int_\theta p_\theta(\boldsymbol{x} \mid y) p(\theta \mid \mathcal{D}) d\theta. \tag{5}$$

Its empirical estimation over an ensemble of M conditional diffusion models is computed as:

$$\widehat{\mathbb{E}}_\theta[p_\theta(\boldsymbol{x} \mid y)] = \frac{1}{M} \sum_{i=1}^{M} p_{\theta_i}(\boldsymbol{x} \mid y).$$

By Equation (5), we recognize that the uncertainty arises from two sources: uncertainty in deciding parameter θ from its posterior $p(\theta|\mathcal{D})$ and uncertainty in generating sample $\boldsymbol{x}$ from a fixed diffusion model $p_\theta(\boldsymbol{x} \mid y)$ after θ is chosen. Before proceeding with the uncertainty decomposition in Diff-BBO, it is crucial to understand how to capture the overall uncertainty when using a diffusion model to generate $\boldsymbol{x}$. Essentially, it can be explicitly traced through the denoising process. More specifically, Theorem 1 provides analytical solutions to compute the uncertainty on a single denoising process of general score-based conditional diffusional models. It offers theoretical insights of how uncertainty is being propagated through the reverse denoising process both in discrete time and continuous time, which is characterized through the lens of stochastic differential equations (SDEs) of the Ornstein–Uhlenbeck (OU) process. Detailed proofs can be found in Appendix A.

Theorem 1. *(Uncertainty propagation) Let $t \in [T]$ be the diffusion step, $s_\theta(\boldsymbol{x}, y, t)$ be the score function of the corresponding diffusion model $p_\theta(\boldsymbol{x} \mid y)$. For a single conditional diffusional model $p_\theta(\boldsymbol{x} \mid y)$, the uncertainty in generating a sample $\boldsymbol{x}$ can be analytically traced through the discrete-time reverse denoising process as follows:*

$$\mathrm{Var}(\boldsymbol{x}_{t-1}) = \tfrac{1}{4}\mathrm{Var}(\boldsymbol{x}_t) + \mathrm{Var}(s_\theta(\boldsymbol{x}, y, t)) + \tfrac{1}{2}\left(\mathbb{E}\left[\boldsymbol{x}_t \circ s_\theta(\boldsymbol{x}_t, y, t)\right] - \mathbb{E}[\boldsymbol{x}_t] \circ \mathbb{E}[s_\theta(\boldsymbol{x}_t, y, t)]\right) + I,$$

$$\mathbb{E}(\boldsymbol{x}_{t-1}) = \frac{1}{2}\mathbb{E}(\boldsymbol{x}_t) + \mathbb{E}(s_\theta(\boldsymbol{x}, y, t)),$$

where $\circ$ is the Hadamard product, and I is the identity matrix. Similarly, in continuous-time process, the uncertainty can be captured as follows:

$$\mathrm{Var}(\boldsymbol{x}_0) = (T+1)I + \mathrm{Var}\left(\int_{t=0}^{T} \left(\frac{1}{2}\boldsymbol{x}_t + s_\theta(\boldsymbol{x}, y, t)\right) \mathrm{d}t\right). \tag{6}$$

While Theorem 1 establishes the existence of closed-form solutions to quantify uncertainty based on the intrinsic properties of diffusion models, performing exact Bayesian inference when training diffusion models in practice requires non-trivial efforts and can be computationally demanding. Hence, in Section 5.2, we will introduce a practically-efficient approach to quantify and decompose the uncertainty based on Equation (5) .

5.2 Uncertainty Decomposition

To systematically analyze the effect of uncertainty in our inverse modeling approach using Diff-BBO, we now provide a practical method to perform uncertainty decomposition in terms of the aleatoric component and its epistemic counterpart.

The *aleatoric uncertainty* in inverse modeling is captured by the variance of the likelihood $p_\theta(\boldsymbol{x} \mid y)$, which is proportional to the variance of the measurement noise during sample generation, irreducible and task-inherent. To estimate the aleatoric uncertainty, we can Monte Carlo (MC) sample $\boldsymbol{x}$ for N times from a learned likelihood function $p_\theta(\boldsymbol{x} \mid y)$ for fixed y, θ.

In contrast, the *epistemic uncertainty* is captured through the variance of the posterior distribution $p(\theta \mid \mathcal{D})$, which is proportional to the variance of the score network, and is reducible with the increase of training data. Recall that Θ is the parameter space that contains all possible model parameters θ, which are used to generate samples from the predictive distribution $p(\boldsymbol{x} \mid y, \mathcal{D})$. As the dataset size and quality grows, the variance of the posterior distribution shrinks, corresponding to the reduction of epistemic uncertainty in learned parameters $\theta \sim p(\theta \mid \mathcal{D})$.

To estimate the epistemic uncertainty, we use ensemble techniques. During the inference time, by initializing the trained ensemble models with different random seeds, we first sample M model parameters $\{\theta_i\}_{i=1}^M$ to simulate M conditional diffusion models. Then we generate N samples $\{\boldsymbol{x}_j\}_{j=1}^N$ for each diffusion model with corresponding parameter $\theta_i,\ \forall i \in [M]$. Combining the above gives a systematic way to decompose and estimate the two types of uncertainty in practice, which is formally described in Proposition 1.

Proposition 1 (Uncertainty Decomposition)**.** *At each iteration $k \in [K]$, the overall uncertainty in inverse modeling can be decomposed into its aleatoric and epistemic components, which can be empirically measured as follows:*

$$\begin{aligned}
\Delta_{\textit{aleatoric}}\,(y, \mathcal{D}) &= \mathbb{E}_{\theta_i \sim p(\cdot|\mathcal{D})}\left[\mathrm{Var}_{\boldsymbol{x}_{i,j} \sim p_{\theta_i}(\cdot|y)}\left(\|\boldsymbol{x}_{i,j}\|\right)\right], \quad \forall i \in [M], j \in [N]; \\
\Delta_{\textit{epistemic}}\,(y, \mathcal{D}) &= \mathrm{Var}_{\theta_i \sim p(\cdot|\mathcal{D})}\left(\mathbb{E}_{\boldsymbol{x}_{i,j} \sim p_{\theta_i}(\cdot|y)}\left[\|\boldsymbol{x}_{i,j}\|\right]\right), \quad \forall i \in [M], j \in [N].
\end{aligned} \tag{7}$$

5.3 Uncertainty-aware Exploration.

At each iteration $k \in [K]$ of Diff-BBO algorithm, the acquisition function $\alpha(y, \mathcal{D})$ proposes an optimal scalar value y_k^* as follows:

$$y_k^* = \mathrm{argmax}_y\, \alpha(y, \mathcal{D}),$$

which is used to generate $\boldsymbol{x}$ in the design space using conditional difussion model.

Note that to design an effective acquisition function for inverse modeling, we need to achieve a balance between high objective values y and low epistemic uncertainty. On the one hand, it is advantageous to focus on the regions in $\mathcal{X}$ whose corresponding y is of high values. As function evaluations are expensive to perform, we prefer to generate samples $\boldsymbol{x}$ conditioned on higher y, and only query the oracle for such promising samples to solve the black-box optimization task. On the other hand, we employ the epistemic uncertainty to gauge the error in the trained diffusion model. Specifically, it helps reduce the approximation error between y_k^* and the reconstructed function value $\max_{j\in[N]} f(\boldsymbol{x}_j)$, where $f(\cdot)$ is the black-box oracle, and $\boldsymbol{x}_j \sim p_\theta(\cdot|y_k^*, \mathcal{D}), \forall j \in [N]$.

We introduce the *Uncertainty-aware Exploration* (UaE) as our designed acquisition function:

$$\alpha(y, \mathcal{D}) = y - \Delta_{\text{epistemic}}(y, \mathcal{D}), \tag{8}$$

which utilizes the uncertainty estimation on conditional diffusion model as given in Proposition 1. It effectively penalizes the candidates for which the model is less certain. As shown later, by balancing the exploration-exploitation trade-off, UaE provides an effective way to solve the online BBO problem.

5.4 Sub-optimality of UaE

To quantify the quality of generated samples, we theoretically analyze the sub-optimality performance gap between y_k^* and reconstructed value at each iteration. In particular, Theorem 2 and Theorem 3 demonstrate that such sub-optimality gap can be effectively handled in inverse modeling, with proofs deferred to Appendix B. We first show that by using conditional diffusion model, the expected error of the sub-optimality performance gap can be effectively bounded under mild assumptions.

Theorem 2. *At each iteration $k \in [K]$, define the sub-optimality performance gap as*

$$\Delta(p_\theta, y_k^*) = \left| y_k^* - \max_{j\in[N]} f(\boldsymbol{x}_j) \right|, \quad \text{where } \ \boldsymbol{x}_j \sim p_\theta(\cdot|y_k^*, \mathcal{D}), \ \ \forall j \in [N]. \tag{9}$$

Assume that there exists some $\theta^* \sim p(\theta|\mathcal{D})$ that produces a probability distribution $p_{\theta^*}(\cdot \mid \mathcal{D})$ such that it is able to generate a sample $\boldsymbol{x}^*$ that perfectly reconstructs y_k^*. Suppose function f is L-Lipschitz and each sample is σ-subGaussian, it can be shown that

$$\mathbb{E}\left[\Delta(p_\theta, y_k^*)\right] \le c_1 L \sqrt{d}\sigma,$$

where d is the dimensionality of the design space, c_1 is some universal constant.

Theorem 2 suggests that in expectation, the reconstructed function value $\max_{j\in[N]} f(\boldsymbol{x}_j)$ closely approximates the provided conditional information y_k^*, implying Diff-BBO is effective in searching for promising samples in the design space by utilizing the information from the objective space. Hence, to achieve a robust estimator for the online BBO problem, the primary concern shifts to controlling the variance of the sub-optimality gap defined in Equation (9), which is further assessed and evaluated in Theorem 3.

Theorem 3. *(Sub-optimality bound) At each iteration $k \in [K]$, suppose M model parameters $\{\theta_i\}_{i=1}^M$ are generated from the ensemble model for some fixed dataset $\mathcal{D}$. Suppose function f is L-Lipschitz, it can be shown that the variance of the sub-optimality performance gap of each model is bounded by the epidemic uncertainty:*

$$\mathrm{Var}\left(\Delta(p_{\theta_i}, y_k^*)\right) \leq c_2 L^2 d\sigma^2 + c_2 L^2 \Delta_{\text{epistemic}}(y_k^*, \mathcal{D}), \ \ \forall i \in M, \tag{10}$$

where c_2 is some universal positive constant.

Theorem 3 shows that the variance of the sub-optimality performance gap can be upper bounded by the epistemic uncertainty of diffusion model with some global constants. It implies that decreasing the epistemic uncertainty will reduce the variance of the performance gap, leading to more reliable optimization performance. Therefore, it is crucial to achieve an effective balance between maximizing the objective value and minimizing epistemic uncertainty when designing the acquisition function. By dosing so, UaE not only explores the objective space with high-value solutions, but also ensures stability and consistency in the optimization process.

Finally, we prove in Theorem 4 that by adopting UaE for inverse modeling to guide the selection of generated samples for solving BBO problems, we can obtain a near-optimal solution for the online optimization problem defined in Equation (3). The proof is available in Appendix C.

Theorem 4. *Let $\mathcal{Y}$ be the constructed candidate set at each iteration $k \in [K]$ in Algorithm 1. By adopting UaE as the acquisition function to guide the sample generation process in conditional diffusion model, Diff-BBO (Algorithm 1) achieves a near-optimal solution for the online BBO problem defined in Equation (3):*

$$\max_{y_k \in \mathbb{R}} \sum_{k=1}^{K} f(\boldsymbol{x}_k), \ \ \boldsymbol{x}_k \sim p_\theta(\cdot \mid y_k, \mathcal{D}), \ \ \theta \in \Theta \ \ \Rightarrow \max_{y_k \in \mathcal{Y}} \sum_{k=1}^{K} \alpha(y_k, \mathcal{D}).$$

As a result, equipped with the novel design of UaE, Diff-BBO is a theoretically sound approach utilizing inverse modeling to effectievely solve the online BBO problem.

6 Experiments

To validate the efficacy of Diff-BBO, we conduct experiments on six online black-box optimization tasks for both continuous and discrete optimization settings. Ablation studies are performed to verify the effectiveness of the proposed acquisition function, assess the robustness of our model in relation to the batch size, and evaluate the computational efficiency of our model. More details of the experimental setups are provided in Appendix D.

6.1 Dataset

We restructured 5 high-dimensional real-world tasks from Design-Bench to facilitate online black-box optimization. We test on 3 continuous and 2 discrete tasks. In **D'Kitty** and **Ant** Morphology, the goal is to optimize for the morphology of robots. In **Superconductor**, the aim is to optimize for finding a superconducting material with a high critical temperature. **TFBind8** and **TFBind10** are discrete tasks where the goal is to find a DNA sequence that has a maximum affinity to bind with a specified transcription factor. We also include a **Molecular Discovery** task to optimize a compound's activity against a biological target with therapeutic value. For each task, we arrange the offline dataset from (Krishnamoorthy et al., 2023) in ascending order based on objective values and select data from the 25th to the 50th percentile as the initial training dataset. We prioritize data with lower objective scores to better observe performance differences across each baseline. Each optimization iteration is allocated 100 queries to the oracle function (batch size $N = 100$), with a total of 16 iterations conducted. More details of the dataset are provided in Appendix D.1.

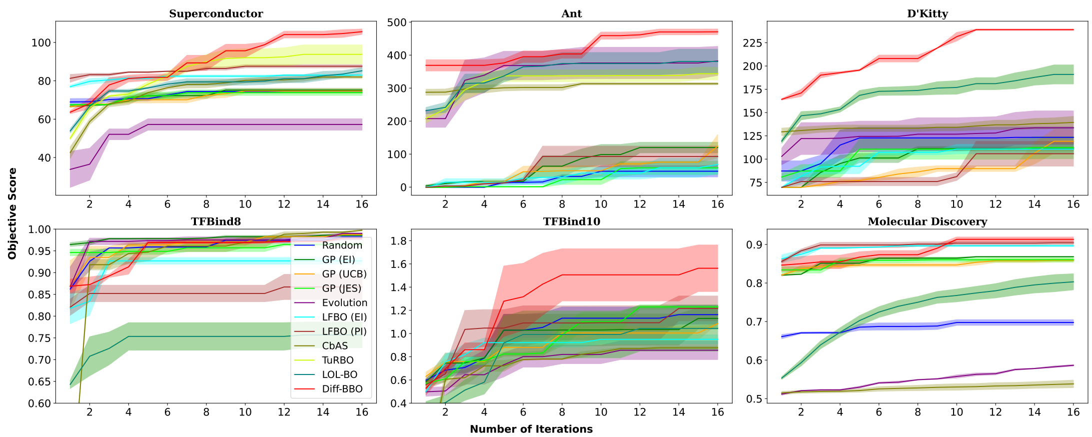

Figure 3: Comparison of Diff-BBO with baselines for online black-box optimization on DesignBench and Molecular Discovery task. All plots start at iteration 1 after one round of data queries. We plot the mean values and the confidence interval based on three random runs. Diff-BBO demonstrates superior performance with few queries to the oracle.

6.2 Baselines

We compare Diff-BBO with 10 baselines, including Bayesian optimization (BO), trust region BO (TuRBO) (Eriksson et al., 2019), local latent space Bayesian optimization (LOL-BO) (Maus et al., 2022), likelihood-free BO (LFBO) (Song et al., 2022), evolutionary algorithms (Brindle, 1980; Real et al., 2019), conditioning by adaptive sampling (CbAS) (Brookes et al., 2019), and random sampling. For BO approaches, we include Gaussian Processes (GP) with Monte Carlo (MC)-based batch expected improvement (EI), MC-based batch upper confidence bound (UCB) (Wilson et al., 2017), and joint entropy search (JES (Hvarfner et al., 2022) as the acquisition functions. For LFBO, we use EI and probability of improvement (PI) as the acquisition functions.

6.3 Results

Figure 3 illustrates the performance across six datasets for all baselines and our proposed algorithm. Notably, Diff-BBO consistently outperforms other baselines in both discrete and continuous settings, with the sole exception of the TF-BIND-8 task. Specifically, in the Ant and Dkitty tasks, Diff-BBO demonstrates a significant lead over all baseline methods, starting from the very first iteration of the online optimization process. This remarkable performance can be attributed to Diff-BBO's diffusion model-based inverse modeling approach, which effectively learns the data manifold in the design space from the initial dataset, even when the initial dataset lacks data with high objective function values. In contrast, the forward approach employed by BO and LFBO, which relies solely on optimizing the trained surrogate model, is more prone to converging on suboptimal solutions.

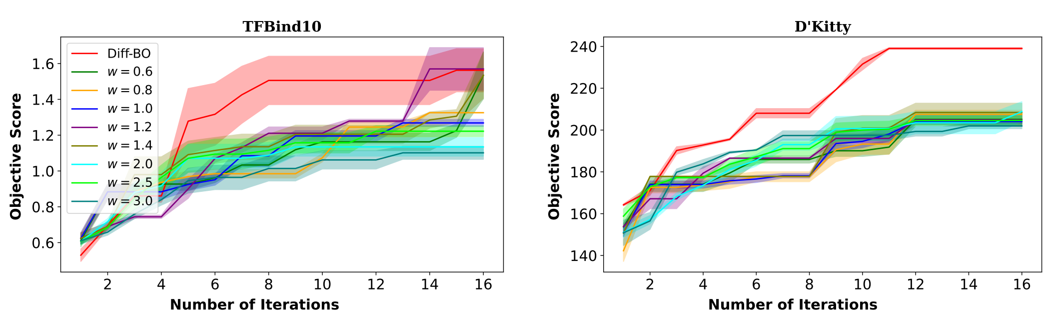

Figure 4: Impact of acquisition function design for black-box optimization on both discrete task (**TFBind10**) and continous task (**D'Kitty**). Comparison of Diff-BBO against fixed-condition approaches using weights $w \in \{0.6, 0.8, 1.0, 1.2, 1.4, 2.0, 2.5, 3.0\}$. Results averaged across three random runs.

Task	GP (EI)	GP (UCB)	Evolution	LFBO (EI)	LFBO (PI)	CbAS	TuRBO	Diff-BBO
TFBind8	112.92	113.81	0.0021	0.59	1.44	0.075	67.23	136.29
Molecular Discovery	53.14	53.82	0.0024	1.93	1.12	0.023	76.84	69.44

Table 1: Model training and acquisition function computation time in seconds.

6.4 Ablation Study

In this section, we conduct ablation studies to investigate the impact of our designed acquisition function, UaE. We compare Diff-BBO with the fixed-condition approach. Instead of using UaE to dynamically determine which $y = w \cdot \phi_k$ to condition on, the fixed condition approach always generates new samples conditioned on $w \cdot \phi_k$ with a fixed weight w. As shown in Figure 4, Diff-BBO consistently outperforms the fixed condition approach. Furthermore, it can be found that simply conditioning on higher y by increasing w does not enhance optimization performance. This highlights the effectiveness of UaE in identifying the optimal y for conditioning by balancing between targeting higher objective values and minimizing the epistemic uncertainty.

Furthermore, we evaluate the effect of batch size, aka the number of queries per iteration on Diff-BBO on the Superconductor task. As shown in Figure 5, we compare the objective function score over number of function evaluations. We can see the performance of our approach remains similar when the batch size becomes larger, suggesting remarkable robustness across different batch sizes. Hence, Diff-BBO is a highly-scalable inverse modeling approach that can efficiently leverage parallelism to handle larger computational loads without compromising performance.

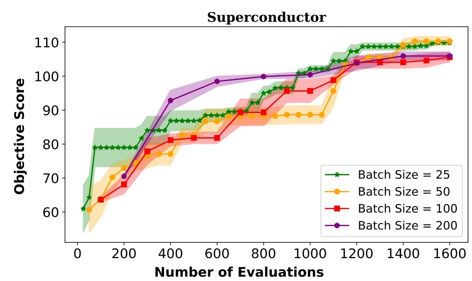

Figure 5: Ablation study to evaluate the effect of batch size on the superconductor task. The mean and standard deviation across three random seeds are plotted. Diff-BBO shows robust performances across different batch size given the same total number of evaluations.

Finally, we analyze the computational time for model training and acquisition function computation for Diff-BBO and existing baselines, as shown in Table 1. The results indicate that the computational time for Diff-BBO is comparable to BO approaches using GP as the surrogate model, typically ranging from 1 to 2 minutes per iteration for model training and acquisition function computation. Given the context of online BBO, where querying the oracle to generate new data is the most expensive or time-consuming part, a few minutes spent on training and acquisition function computation should not be considered a significant computational burden.

7 Conclusion, Limitation, and Future Work

In this paper, we introduced Diff-BBO, a novel inverse modeling approach for black-box optimization that leverages the uncertainty of conditional diffusion models. By utilizing the novel acquisition function UaE, Diff-BBO strategically proposes objective function values to improve sample efficiency in online settings. Our empirical evaluations on the Design-Bench benchmark and molecular design experiments demonstrate that Diff-BBO achieves state-of-the-art performance, establishing its potential as a robust tool for efficient and effective online black-box optimization. Theoretically, we prove that using UaE leads to optimal optimization solutions. We conclude by discussing the limitations and potential extensions of Diff-BBO: (i) *acquisition function improvement:* Our current implementation for the acquisition function, UaE, requires presetting the candidate sets. This necessitates additional hyperparameter tuning. (ii) *BBO extensions:* Diff-BBO can be extended to various Bayesian optimization settings, including multi-objective and multi-fidelity Bayesian optimization.

Reproducibility Statement

The experimental details can be found in Appendix D. We provide the code in the supplementary materials for reproducing the results.

REFERENCES

Shipra Agrawal and Navin Goyal. Analysis of thompson sampling for the multi-armed bandit problem. In *Conference on learning theory*, pp. 39–1. JMLR Workshop and Conference Proceedings, 2012.

Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Soumyadip Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Universal guidance for diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 843–852, 2023.

Anne Brindle. Genetic algorithms for function optimization. 1980.

David Brookes, Hahnbeom Park, and Jennifer Listgarten. Conditioning by adaptive sampling for robust design. In *International conference on machine learning*, pp. 773–782. PMLR, 2019.

Matthew A Chan, Maria J Molina, and Christopher A Metzler. Hyper-diffusion: Estimating epistemic and aleatoric uncertainty with a single model. *arXiv preprint arXiv:2402.03478*, 2024.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Zhekai Du and Jingjing Li. Diffusion-based probabilistic uncertainty estimation for active domain adaptation. *Advances in Neural Information Processing Systems*, 36:17129–17155, 2023.

Peter Eckmann, Kunyang Sun, Bo Zhao, Mudong Feng, Michael K Gilson, and Rose Yu. Limo: Latent inceptionism for targeted molecule generation. *arXiv preprint arXiv:2206.09010*, 2022.

Canberk Ekmekci and Mujdat Cetin. Quantifying generative model uncertainty in posterior sampling methods for computational imaging. In *NeurIPS 2023 Workshop on Deep Learning and Inverse Problems*, 2023.

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

Peter I Frazier. A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.

Justin Fu and Sergey Levine. Offline model-based optimization via normalized maximum likelihood estimation. *arXiv preprint arXiv:2102.07970*, 2021.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.

Nate Gruver, Samuel Stanton, Nathan Frey, Tim GJ Rudner, Isidro Hotzel, Julien Lafrance-Vanasse, Arvind Rajpal, Kyunghyun Cho, and Andrew G Wilson. Protein design with guided discrete diffusion. *Advances in neural information processing systems*, 36, 2024.

Ali Hebbal, Loic Brevault, Mathieu Balesdent, El-Ghazali Talbi, and Nouredine Melab. Bayesian optimization using deep gaussian processes. *arXiv preprint arXiv:1905.03350*, 2019.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Carl Hvarfner, Frank Hutter, and Luigi Nardi. Joint entropy search for maximally-informed bayesian optimization. *Advances in Neural Information Processing Systems*, 35:11494–11506, 2022.

Woosung Jeon and Dongsup Kim. Autonomous molecule generation using reinforcement learning and docking to develop potential novel inhibitors. *Scientific reports*, 10(1):22104, 2020.

Amin Karbasi, Nikki Lijing Kuang, Yian Ma, and Siddharth Mitra. Langevin thompson sampling with logarithmic communication: bandits and reinforcement learning. In *International Conference on Machine Learning*, pp. 15828–15860. PMLR, 2023.

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems*, 30, 2017.

Minsu Kim, Federico Berto, Sungsoo Ahn, and Jinkyoo Park. Bootstrapped training of score-conditioned generator for offline design of biological sequences. *Advances in Neural Information Processing Systems*, 37, 2023.

Lingkai Kong, Yuanqi Du, Wenhao Mu, Kirill Neklyudov, Valentin De Bortol, Haorui Wang, Dongxia Wu, Aaron Ferber, Yi-An Ma, Carla P Gomes, et al. Diffusion models as constrained samplers for optimization with unknown constraints. *arXiv preprint arXiv:2402.18012*, 2024.

Siddarth Krishnamoorthy, Satvik Mehul Mashkaria, and Aditya Grover. Diffusion models for black-box optimization. In *International Conference on Machine Learning*, pp. 17842–17857. PMLR, 2023.

Aviral Kumar and Sergey Levine. Model inversion networks for model-based optimization. *Advances in neural information processing systems*, 33:5126–5137, 2020.

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

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

Seul Lee, Jaehyeong Jo, and Sung Ju Hwang. Exploring chemical space with score-based out-of-distribution generation. In *International Conference on Machine Learning*, pp. 18872–18892. PMLR, 2023.

Zihao Li, Hui Yuan, Kaixuan Huang, Chengzhuo Ni, Yinyu Ye, Minshuo Chen, and Mengdi Wang. Diffusion model for data-driven black-box optimization. *arXiv preprint arXiv:2403.13219*, 2024.

Tennison Liu, Nicolás Astorga, Nabeel Seedat, and Mihaela van der Schaar. Large language models to enhance bayesian optimization. *arXiv preprint arXiv:2402.03921*, 2024.

Huakang Lu, Hong Qian, Yupeng Wu, Ziqi Liu, Ya-Lin Zhang, Aimin Zhou, and Yang Yu. Degradation-resistant offline optimization via accumulative risk control. In *ECAI 2023*, pp. 1609–1616. IOS Press, 2023.

David JC MacKay. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472, 1992.

Natalie Maus, Haydn Jones, Juston Moore, Matt J Kusner, John Bradshaw, and Jacob Gardner. Local latent space bayesian optimization over structured inputs. *Advances in neural information processing systems*, 35:34505–34518, 2022.

Jonas Mockus. On bayesian methods for seeking the extremum. In *Proceedings of the IFIP Technical Conference*, pp. 400–404, 1974.

Garrett M Morris, Ruth Huey, William Lindstrom, Michel F Sanner, Richard K Belew, David S Goodsell, and Arthur J Olson. Autodock4 and autodocktools4: Automated docking with selective receptor flexibility. *Journal of computational chemistry*, 30(16):2785–2791, 2009.

Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

Juhwan Noh, Dae-Woong Jeong, Kiyoung Kim, Sehui Han, Moontae Lee, Honglak Lee, and Yousung Jung. Path-aware and structure-preserving generation of synthetically accessible molecules. In *International Conference on Machine Learning*, pp. 16952–16968. PMLR, 2022.

Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. In *Proceedings of the aaai conference on artificial intelligence*, volume 33, pp. 4780–4789, 2019.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Benjamin Sanchez-Lengeling and Alán Aspuru-Guzik. Inverse molecular design using machine learning: Generative models for matter engineering. *Science*, 361(6400):360–365, 2018.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. PMLR, 2015.

Jiaming Song, Lantao Yu, Willie Neiswanger, and Stefano Ermon. A general recipe for likelihood-free bayesian optimization. In *International Conference on Machine Learning*, pp. 20384–20404. PMLR, 2022.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

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

Matthew Tesch, Jeff Schneider, and Howie Choset. Expensive multiobjective optimization for robotics. In *2013 ieee international conference on robotics and automation*, pp. 973–980. IEEE, 2013.

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*, pp. 5026–5033. IEEE, 2012.

Brandon Trabucco, Xinyang Geng, Aviral Kumar, and Sergey Levine. Design-bench: Benchmarks for data-driven offline model-based optimization. In *International Conference on Machine Learning*, pp. 21658–21676. PMLR, 2022.

Ryan Turner, David Eriksson, Michael McCourt, Juha Kiili, Eero Laaksonen, Zhen Xu, and Isabelle Guyon. Bayesian optimization is superior to random search for machine learning hyperparameter tuning: Analysis of the black-box optimization challenge 2020. In *NeurIPS 2020 Competition and Demonstration Track*, pp. 3–26. PMLR, 2021.

Matias Valdenegro-Toro and Daniel Saromo Mori. A deeper look into aleatoric and epistemic uncertainty disentanglement. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1508–1516. IEEE, 2022.

Martin J Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge university press, 2019.

Handing Wang, Yaochu Jin, Chaoli Sun, and John Doherty. Offline data-driven evolutionary optimization using selective surrogate ensembles. *IEEE Transactions on Evolutionary Computation*, 23(2):203–216, 2018.

Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*, 2022.

James T Wilson, Riccardo Moriconi, Frank Hutter, and Marc Peter Deisenroth. The reparameterization trick for acquisition functions. *arXiv preprint arXiv:1712.00424*, 2017.

Dongxia Wu, Ruijia Niu, Matteo Chinazzi, Yian Ma, and Rose Yu. Disentangled multi-fidelity deep bayesian active learning. In *International Conference on Machine Learning*, pp. 37624–37634. PMLR, 2023.

Cheng Zhang, Judith Bütepage, Hedvig Kjellström, and Stephan Mandt. Advances in variational inference. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):2008–2026, 2018.

Dinghuai Zhang, Jie Fu, Yoshua Bengio, and Aaron Courville. Unifying likelihood-free inference with black-box optimization and beyond. *arXiv preprint arXiv:2110.03372*, 2021.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3836–3847, 2023.

Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

Appendices

A Uncertainty Quantification Through SDEs

A.1 Conditional Diffusion SDE

It can be shown that the conditional diffusion model can be represented by the Ornstein–Uhlenbeck (OU) process, which is a time-homogeneous continuous-time Markov process:

$$\mathrm{d}\boldsymbol{x}_t = -\gamma \boldsymbol{x}_t \,\mathrm{d}t + \sigma \,\mathrm{d}\boldsymbol{w}_t, \tag{11}$$

where γ is the relaxation rate, σ is the strength of fluctuation, and $\boldsymbol{w}_t$ is the standard Wiener process (a.k.a., Brownian motion). Both γ and σ are time-invariant. In particular, setting $\gamma = 1$ and $\sigma = \sqrt{2}$, we are able to establish that Denoising Diffusion Probabilistic Model (DDPM) is equivalent to OU process observed at discrete times. In the remaining text, we consider SDEs for general score-based diffusion models. The SDE of the forward process in conditional diffusion model can then be written as:

$$\mathrm{d}\boldsymbol{x}_t = -\frac{1}{2} g(t)\boldsymbol{x}_t \,\mathrm{d}t + \sqrt{g(t)}\,\mathrm{d}\boldsymbol{w}_t, \;\; \boldsymbol{x}_0 \sim q(\boldsymbol{x}|y) \tag{12}$$

where $g(t)$ is a nondecreasing weighting function that controls the speed of diffusion in the forward process and $g(t) > 0$. For simplicity of analysis, we fix $g(t) = 1$ for all $t \in [T]$.

The generation process of a conditional score-based diffusion model can be viewed as a particular discretization of the following reverse-time SDE:

$$\mathrm{d}\boldsymbol{x}_t = \left(\frac{1}{2}\boldsymbol{x}_t - \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|y)\right) \mathrm{d}t + \mathrm{d}\boldsymbol{w}_t, \;\; \boldsymbol{x}_0 \sim p(\boldsymbol{x}_T|y). \tag{13}$$

In practice, the unknown ground truth conditional score $\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|y)$ needs to be estimated with score networks. Let such estimator denoted by $s_\theta(\boldsymbol{x}, y, t)$, then the conditional sample generation is to simulate the following backward SDE:

$$\mathrm{d}\boldsymbol{x}_t = \left(\frac{1}{2}\boldsymbol{x}_t - s_\theta(\boldsymbol{x}, y, t)\right) \mathrm{d}t + \mathrm{d}\boldsymbol{w}_t, \;\; \boldsymbol{x}_0 \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}). \tag{14}$$

In Bayesian settings, we sample a score function $\widetilde{s}_\theta(\boldsymbol{x}_t, y, t)$ from the probability distribution $p(s_\theta|\boldsymbol{x}_t, y, t, \mathcal{D}) = \mathcal{N}(s_\theta(\boldsymbol{x}_t, y, t), \Sigma_\theta(\boldsymbol{x}_t, y, t))$ with expected value $s_\theta(\boldsymbol{x}_t, y, t)$, and diagonal covariance $\Sigma_\theta(\boldsymbol{x}_t, y, t)$.

A.2 Estimation of Uncertainty

In this section, we quantify the uncertainty of a single conditional diffusion model in both discrete-time and continuous-time reverse process for Theorem 1.

A.2.1 Uncertainty in Discrete-time Reverse Process

We first proof the first statement of Theorem 1. We consider the Euler discretization of Equation (14), which leads to:

$$\boldsymbol{x}_{t-1} = \frac{1}{2}\boldsymbol{x}_t + s_\theta(\boldsymbol{x}, y, t) + \boldsymbol{\epsilon}, \;\; \epsilon \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}). \tag{15}$$

We thus have,

$$\mathrm{Var}(\boldsymbol{x}_{t-1}) = \frac{1}{4}\mathrm{Var}(\boldsymbol{x}_t) + \mathrm{Var}(s_\theta(\boldsymbol{x}, y, t)) + \frac{1}{2}\mathrm{Cov}\left(\boldsymbol{x}_t, s_\theta(\boldsymbol{x}, y, t)\right) + I. \tag{16}$$

$$\mathbb{E}(\boldsymbol{x}_{t-1}) = \frac{1}{2}\mathbb{E}(\boldsymbol{x}_t) + \mathbb{E}(s_\theta(\boldsymbol{x}, y, t)). \tag{17}$$

Here $\mathrm{Cov}\left(\boldsymbol{x}_t, s_\theta(\boldsymbol{x}, y, t)\right)$ is the element-vise covariance between $\boldsymbol{x}_t$ and $s_\theta(\boldsymbol{x}, y, t)$. Note that we only need to consider the correlation between $\boldsymbol{x}_t$ and $s_\theta(\boldsymbol{x}, y, t)$ at the same time step. As a result, to estimate $\mathrm{Cov}\left(\boldsymbol{x}_t, s_\theta(\boldsymbol{x}, y, t)\right)$, we have,

$$\begin{aligned}
\mathrm{Cov}\left(\boldsymbol{x}_t, s_\theta(\boldsymbol{x}, y, t)\right) &= \mathbb{E}\left[\left(\boldsymbol{x}_t - \mathbb{E}[\boldsymbol{x}_t]\right)\left(s_\theta(\boldsymbol{x}, y, t) - \mathbb{E}[s_\theta(\boldsymbol{x}, y, t)]\right)^{\mathrm{T}}\right] \\
&= \mathbb{E}\left[\boldsymbol{x}_t \circ s_\theta(\boldsymbol{x}, y, t)\right] - \mathbb{E}[\boldsymbol{x}_t] \circ \mathbb{E}[s_\theta(\boldsymbol{x}, y, t)] \\
&= \mathbb{E}_{\boldsymbol{x}_t}\left[\boldsymbol{x}_t \circ s_\theta(\boldsymbol{x}, y, t)\right] - \mathbb{E}[\boldsymbol{x}_t] \circ \mathbb{E}_{\boldsymbol{x}_t}[s_\theta(\boldsymbol{x}_t, y, t)]
\end{aligned} \tag{18}$$

where $\circ$ is the Hadamard product and the third equality is by tower's rule. Substituting Equation (18) back to Equation (16) completes the proof of the first part of Theorem 1.

A.2.2 Uncertainty in Continuous-time Reverse Process

We now proof the second statement of Theorem 1. To perform the uncertainty quantification for the continuous-time reverse process, we posit the following assumption.

Assumption 1. *For valid $t \in [0, T]$, the generating process $\boldsymbol{x}_t$ in Equation (13) is integrable and has finite second-order moments.*

With Assumption 1, integrating Equation (13) with respect to t yields:

$$\boldsymbol{x}_0 = \boldsymbol{x}_T - \int_{t=0}^{T} \left(\frac{1}{2}\boldsymbol{x}_t + \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|y) \right) \mathrm{d}t + \int_{t=0}^{T} \mathrm{d}\boldsymbol{w}_t. \tag{19}$$

Applying the variance operator to both sides of

$$\begin{aligned} \mathrm{Var}(\boldsymbol{x}_0) &= \mathrm{Var}(\boldsymbol{x}_T) + \mathrm{Var}\left(\int_{t=0}^{T} \left(\frac{1}{2}\boldsymbol{x}_t + \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|y) \right) \mathrm{d}t \right) + \mathrm{Var}\left(\int_{t=0}^{T} \mathrm{d}\boldsymbol{w}_t \right) \\ &= I + \mathrm{Var}\left(\int_{t=0}^{T} \left(\frac{1}{2}\boldsymbol{x}_t + \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|y) \right) \mathrm{d}t \right) + \mathbb{E}\left[\left(\int_{t=0}^{T} \mathrm{d}\boldsymbol{w}_t \right)^2 \right] - \left(\mathbb{E}\left[\int_{t=0}^{T} \mathrm{d}\boldsymbol{w}_t \right] \right)^2 \\ &= (T+1)I + \underbrace{\mathrm{Var}\left(\int_{t=0}^{T} \left(\frac{1}{2}\boldsymbol{x}_t + \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|y) \right) \mathrm{d}t \right)}_{V_1}, \end{aligned} \tag{20}$$

where the last equality follows the properties of Itô Integral and rules of stochastic calculus such that $(\mathrm{d}\boldsymbol{w})^2 = \mathrm{d}t$, $\mathbb{E}[\int_{t=0}^{T} \mathrm{d}\boldsymbol{w}_t] = 0$. Hence, to provide an uncertainty estimate for $\boldsymbol{x}_0$, it remains to estimate the term V_1. Recall that the true score function $\nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t|y)$ is approximated by $s_\theta((\boldsymbol{x}_t, y, t) = -\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, y)/\sigma_t$. For ease of notation, let $s_{\theta,t} = s_\theta(\boldsymbol{x}_t, y, t)$ and $\widetilde{s}_{\theta,t} = \widetilde{s}_\theta(\boldsymbol{x}_t, y, t)$, which gives

$$V_1 = \int_{t=0}^{T} \int_{s=0}^{T} \left(\frac{1}{4}\mathrm{Cov}(\boldsymbol{x}_s, \boldsymbol{x}_t) - \frac{1}{2}\mathrm{Cov}(\boldsymbol{x}_s, s_{\theta,t}) - \frac{1}{2}\mathrm{Cov}(\boldsymbol{x}_t, s_{\theta,s}) + \mathrm{Cov}(s_{\theta,t}, s_{\theta,s}) \right) \mathrm{d}s\, \mathrm{d}t.$$

When $s \neq t$, score functions $s_{\theta,t}$ and $s_{\theta,s}$ are independent, and similarly, $\boldsymbol{x}_t$ and $s_{\theta,s}$ are also independent. As a result, the above equation can be further simplified as

$$V_1 = \int_{t=0}^{T} \int_{s=0}^{T} \left(\frac{1}{4}\mathrm{Cov}(\boldsymbol{x}_s, \boldsymbol{x}_t) - \frac{1}{2}\mathrm{Cov}(\boldsymbol{x}_s, s_{\theta,t}) \right) \mathrm{d}s\, \mathrm{d}t - \int_{t=0}^{T} \left(\mathrm{Cov}(\boldsymbol{x}_t, s_{\theta,t}) + \mathrm{Cov}(s_{\theta,t}, s_{\theta,t}) \right) \mathrm{d}t.$$

Combining all the above results together completes the proof of the second statement of Theorem 1.

B Analysis of Sub-optimality for Black-box Function

In this section, we study the behavior of the sub-optimality gap of our algorithm by proving Theorem 2 and Theorem 3. We first introduce the notation that is used throughout this section and the next section. Then we present the main lemmas along with their proofs. Finally, we combine the lemmas to prove our main results.

At each iteration $k \in [K]$, let y_k^* be the target function value on which the diffusion model conditions, and p_θ be the model learned by the conditional diffusion model. We define the performance metric for online BBO problem, which measures the sub-optimal performance gap between the function value achieved by sample $\boldsymbol{x} \sim p_\theta(\cdot|y_k^*, \mathcal{D})$ and the target function value y_k^*. Its formal definition is described as follows:

$$\Delta(p_\theta, y_k^*) = \left| y_k^* - \max_{j \in [N]} f(\boldsymbol{x}_j) \right|, \quad \text{where } \boldsymbol{x}_j \sim p_\theta(\cdot|y_k^*, \mathcal{D}), \ \forall j \in [N]. \tag{21}$$

For simplicity of analysis, we consider $N = 1$, and let the generated sample at the k-th iteration be $\boldsymbol{x}_k$ in the remaining text. We remark that all proofs go through smoothly for general N with more nuanced notations, and do not affect the conclusions being drawn. To proceed with the proofs in this section, we first state the formal assumptions for the black-box function $f(\cdot)$ and sample $\boldsymbol{x}$.

Assumption 2. *The scalar black-box function f is L-Lipschitz in $\boldsymbol{x}$:*

$$|f(\boldsymbol{x}') - f(\boldsymbol{x})| \leq L\|\boldsymbol{x}' - \boldsymbol{x}\|, \ \ \forall \boldsymbol{x}', \boldsymbol{x} \in \mathbb{R}^d.$$

Assumption 3. *Each generated sample $\boldsymbol{x} \in \mathbb{R}^d$ is σ-subGaussian. That is, there exists $\sigma \in \mathbb{R}$ such that for any $\boldsymbol{v} \in \mathbb{R}^d$ with $\|\boldsymbol{v}\| = 1$, $\boldsymbol{v}^{\mathrm{T}}(\boldsymbol{x} - \mathbb{E}[\boldsymbol{x}])$ is σ-subGaussian, and its moment generating function is bounded by:*

$$\mathbb{E}[\exp\left(\lambda \boldsymbol{v}^{\mathrm{T}}(\boldsymbol{x} - \mathbb{E}[\boldsymbol{x}])\right)] \leq \exp\left(\frac{\sigma^2\lambda^2}{2}\right), \quad \forall \lambda \in \mathbb{R}, \ \boldsymbol{v} \in \mathbb{S}^{d-1},$$

where $\mathbb{S} := \{\boldsymbol{v} \in \mathbb{R}^d : \|\boldsymbol{v}\| = 1\}$ is the $(d-1)$ unit sphere.

Before proceeding with the proofs of main theorems, we present our main lemmas.

Lemma B.1. *At each iteration $k \in [K]$, under fixed parameters θ and θ^*, for $\boldsymbol{x}_k \sim p_\theta(\cdot|y_k^*, \mathcal{D})$, $\boldsymbol{x}^* \sim p_{\theta^*}(\cdot|y_k^*, \mathcal{D})$, we have*

$$\mathbb{E}_{\boldsymbol{x}_k \sim p_\theta(\cdot|y_k^*, \mathcal{D}), \boldsymbol{x}^* \sim p_{\theta^*}(\cdot|y_k^*, \mathcal{D})}\left[\|\boldsymbol{x}^* - \boldsymbol{x}_k\|\right] \leq 8\sqrt{d}\sigma + \|\mathbb{E}_{\boldsymbol{x}^*}[\boldsymbol{x}^*] - \mathbb{E}_{\boldsymbol{x}_k}[\boldsymbol{x}_k]\|, \tag{22}$$

$$\mathbb{E}_{\boldsymbol{x}_k \sim p_\theta(\cdot|y_k^*, \mathcal{D}), \boldsymbol{x}^* \sim p_{\theta^*}(\cdot|y_k^*, \mathcal{D})}\left[\|\boldsymbol{x}^* - \boldsymbol{x}_k\|\right] \geq \|\mathbb{E}_{\boldsymbol{x}^*}[\boldsymbol{x}^*] - \mathbb{E}_{\boldsymbol{x}_k}[\boldsymbol{x}_k]\|. \tag{23}$$

Proof of Lemma B.1. To bound $\mathbb{E}\left[\|\boldsymbol{x}^* - \boldsymbol{x}_k\|\right]$, by triangle inequality,

$$\begin{aligned}\mathbb{E}_{\boldsymbol{x}_k, \boldsymbol{x}^*}\left[\|\boldsymbol{x}^* - \boldsymbol{x}_k\|\right] &= \mathbb{E}\left[\|\boldsymbol{x}^* - \mathbb{E}[\boldsymbol{x}^*] + \mathbb{E}[\boldsymbol{x}_k] - \boldsymbol{x}_k + \mathbb{E}[\boldsymbol{x}^*] - \mathbb{E}[\boldsymbol{x}_k]\|\right] \\ &\leq \mathbb{E}\left[\|\boldsymbol{x}^* - \mathbb{E}[\boldsymbol{x}^*]\|\right] + \mathbb{E}\left[\|\boldsymbol{x}_k - \mathbb{E}[\boldsymbol{x}_k]\|\right] + \mathbb{E}\left[\|\mathbb{E}[\boldsymbol{x}^*] - \mathbb{E}[\boldsymbol{x}_k]\|\right].\end{aligned}$$

Under assumption 3, by Lemma B.3, we have,

$$\mathbb{E}_{\boldsymbol{x}_k, \boldsymbol{x}^*}\left[\|\boldsymbol{x}^* - \boldsymbol{x}_k\|\right] \leq 8\sqrt{d}\sigma + \|\mathbb{E}[\boldsymbol{x}^*] - \mathbb{E}[\boldsymbol{x}_k]\|.$$

Applying triangle inequality completes the step. In addition, it can be easily seen that

$$\mathbb{E}_{\boldsymbol{x}_k, \boldsymbol{x}^*}\left[\|\boldsymbol{x}^* - \boldsymbol{x}_k\|\right] \geq \|\mathbb{E}[\boldsymbol{x}^*] - \mathbb{E}[\boldsymbol{x}_k]\|.$$

□

Lemma B.2. *At each iteration $k \in [K]$, under fixed parameters θ and θ^*, for $\boldsymbol{x}_k \sim p_\theta(\cdot|y_k^*, \mathcal{D})$, $\boldsymbol{x}^* \sim p_{\theta^*}(\cdot|y_k^*, \mathcal{D})$, we have*

$$\mathrm{Var}_{\boldsymbol{x}_k \sim p_\theta(\cdot|y_k^*, \mathcal{D}), \boldsymbol{x}^* \sim p_{\theta^*}(\cdot|y_k^*, \mathcal{D})}(\|\boldsymbol{x}^* - \boldsymbol{x}_k\|) \leq c_3 d\sigma^2. \tag{24}$$

Proof of Lemma B.2. By definition of variance,

$$\mathrm{Var}_{\boldsymbol{x}_k, \boldsymbol{x}^*}(\|\boldsymbol{x}^* - \boldsymbol{x}_k\|) = \mathbb{E}[\|\boldsymbol{x}^* - \boldsymbol{x}_k\|^2] - (\mathbb{E}[\|\boldsymbol{x}^* - \boldsymbol{x}_k\|])^2. \tag{25}$$

Expanding the first term leads to

$$\begin{aligned}\mathbb{E}[\|\boldsymbol{x}^* - \boldsymbol{x}_k\|^2] &= \mathbb{E}[(\boldsymbol{x}^* - \boldsymbol{x}_k)^{\mathrm{T}}(\boldsymbol{x}^* - \boldsymbol{x}_k)] \\ &= \mathbb{E}[\|\boldsymbol{x}^*\|^2] + \mathbb{E}[\|\boldsymbol{x}_k\|^2] - 2\mathbb{E}[(\boldsymbol{x}_k)^{\mathrm{T}}\boldsymbol{x}^*] \\ &= \mathbb{E}[\|\boldsymbol{x}^*\|^2] + \mathbb{E}[\|\boldsymbol{x}_k\|^2] - 2\mathbb{E}[(\boldsymbol{x}_k)]^{\mathrm{T}}\mathbb{E}[\boldsymbol{x}^*],\end{aligned} \tag{26}$$

where the last equality is due to the independece between $\boldsymbol{x}^*$ and $\boldsymbol{x}_k$.

Under Assumption 3 and by Lemma B.4, we have

$$\begin{aligned}\mathbb{E}[\|\boldsymbol{x}^*\|^2] &= \mathbb{E}[\|\boldsymbol{x}^* - \mathbb{E}[\boldsymbol{x}^*] + \mathbb{E}[\boldsymbol{x}^*]\|^2] \\ &= \mathbb{E}[(\boldsymbol{x}^* - \mathbb{E}[\boldsymbol{x}^*])^{\mathrm{T}}(\boldsymbol{x}^* - \mathbb{E}[\boldsymbol{x}^*])] + \|\mathbb{E}[\boldsymbol{x}^*]\|^2 \\ &= \mathrm{tr}(\mathbb{E}[(\boldsymbol{x}^* - \mathbb{E}[\boldsymbol{x}^*])(\boldsymbol{x}^* - \mathbb{E}[\boldsymbol{x}^*])]^{\mathrm{T}}) + \|\mathbb{E}[\boldsymbol{x}^*]\|^2 \\ &\leq Cd\sigma^2 + \|\mathbb{E}[\boldsymbol{x}^*]\|^2.\end{aligned}$$

Here, the second equality holds as the cross terms vanish due to the fact that $\mathbb{E}[\boldsymbol{x}^* - \mathbb{E}[\boldsymbol{x}^*]] = 0$. Similarly,

$$\mathbb{E}[\|\boldsymbol{x}_k\|^2] \leq Cd\sigma^2 + \|\mathbb{E}[\boldsymbol{x}_k]\|^2.$$

Substituting the above two results back to Equation (26),

$$\mathbb{E}[\|\boldsymbol{x}^* - \boldsymbol{x}_k\|^2] \leq 2Cd\sigma^2 + \|\mathbb{E}[\boldsymbol{x}_k]\|^2 + \|\mathbb{E}[\boldsymbol{x}^*]\|^2 - 2\mathbb{E}[(\boldsymbol{x}_k)^{\mathrm{T}}\boldsymbol{x}^*]$$
$$\leq 2Cd\sigma^2 + \|\mathbb{E}[\boldsymbol{x}_k] - \mathbb{E}[\boldsymbol{x}^*]\|^2 . \tag{27}$$

Substituting Equation (27) back to Equation (25) and applying Lemma B.1 leads to

$$\mathrm{Var}_{\boldsymbol{x}_k,\boldsymbol{x}^*}(\|\boldsymbol{x}^* - \boldsymbol{x}_k\|) \leq 2Cd\sigma^2 + \|\mathbb{E}[\boldsymbol{x}_k] - \mathbb{E}[\boldsymbol{x}^*]\|^2 - (8\sqrt{d}\sigma + \|\mathbb{E}[\boldsymbol{x}^*] - \mathbb{E}[\boldsymbol{x}_k]\|)^2 \leq c_3 d\sigma^2.$$

□

With the above results, we are ready to prove Theorem 2 and Theorem 3.

Theorem 2. *At each iteration $k \in [K]$, define the sub-optimality performance gap as*

$$\Delta(p_\theta, y_k^*) = \left| y_k^* - \max_{j\in[N]} f(\boldsymbol{x}_j) \right|, \quad \text{where } \boldsymbol{x}_j \sim p_\theta(\cdot|y_k^*, \mathcal{D}), \ \ \forall j \in [N]. \tag{9}$$

Assume that there exists some $\theta^* \sim p(\theta|\mathcal{D})$ that produces a probability distribution $p_{\theta^*}(\cdot \mid \mathcal{D})$ such that it is able to generate a sample $\boldsymbol{x}^*$ that perfectly reconstructs y_k^*. Suppose function f is L-Lipschitz and each sample is σ-subGaussian, it can be shown that

$$\mathbb{E}\left[\Delta(p_\theta, y_k^*)\right] \leq c_1 L\sqrt{d}\sigma,$$

where d is the dimensionality of the design space, c_1 is some universal constant.

Proof of Theorem 2. Recall that we consider the case where $N = 1$, and denote $\boldsymbol{x}_k$ the generated sample in the k-th iteration, i.e. $\boldsymbol{x}_k \sim p_\theta(\cdot|y_k^*, \mathcal{D})$, where $\theta \sim p(\theta \mid \mathcal{D})$. In each iteration k, with the existence of $\theta^* \sim p(\theta \mid \mathcal{D})$, we have $y_k^* = f(\boldsymbol{x}^*)$, where $\boldsymbol{x}^* \sim p_{\theta^*}(\cdot|y_k^*, \mathcal{D})$. Hence, under Assumption 2,

$$\mathbb{E}\left[\Delta(p_\theta, y_k^*)\right] = \mathbb{E}\left[|f(\boldsymbol{x}^*) - f(\boldsymbol{x}_k)|\right] \leq L\mathbb{E}\left[\|\boldsymbol{x}^* - \boldsymbol{x}_k\|\right].$$

By Lemma B.1, tower rule and Lemma B.5, we have

$$\mathbb{E}\left[\Delta(p_\theta, y_k^*)\right] \leq L\mathbb{E}_{\theta,\theta^*}\left[\mathbb{E}_{x,x^*}\left[\|\boldsymbol{x}^* - \boldsymbol{x}_k\|\right] |\theta, \theta^*\right]$$
$$\leq 8L\sqrt{d}\sigma + \mathbb{E}_{\theta,\theta^*}\left[\|\mathbb{E}_{\boldsymbol{x}^*}[\boldsymbol{x}^*|\theta^*] - \mathbb{E}_{\boldsymbol{x}_k}[\boldsymbol{x}_k|\theta]\|\right]$$
$$\leq c_1 L\sqrt{d}\sigma.$$

□

Theorem 3. *(Sub-optimality bound) At each iteration $k \in [K]$, suppose M model parameters $\{\theta_i\}_{i=1}^M$ are generated from the ensemble model for some fixed dataset $\mathcal{D}$. Suppose function f is L-Lipschitz, it can be shown that the variance of the sub-optimality performance gap of each model is bounded by the epidemic uncertainty:*

$$\mathrm{Var}\left(\Delta(p_{\theta_i}, y_k^*)\right) \leq c_2 L^2 d\sigma^2 + c_2 L^2 \Delta_{\text{epistemic}}(y_k^*, \mathcal{D}), \ \ \forall i \in M, \tag{10}$$

where c_2 is some universal positive constant.

Proof of Theorem 3. At every iteration $k \in [K]$, let the target function value on which the conditional diffusion model conditions be y_k^*. The statement needs to hold for each conditional diffusion model in the ensemble, and thus for simplicity of notation, the subscript i of θ_i is dropped in the remaining proof. With the existence of $\theta^* \sim p(\theta \mid \mathcal{D})$, we have $y_k^* = f(\boldsymbol{x}^*)$, where $\boldsymbol{x}^* \sim p_{\theta^*}(\cdot|y_k^*, \mathcal{D})$. Recall that $f(\boldsymbol{x}_k)$ is achieved by $\boldsymbol{x}_k \sim p_\theta(\cdot|y_k^*, \mathcal{D})$, where $\theta \sim p(\theta \mid \mathcal{D})$, and $N = 1$.

Thus, by Eve's law, the overall variance of $\Delta(p_\theta, y_k^*)$ can be decomposed as:

$$\mathrm{Var}\left(\Delta(p_\theta, y_k^*)\right) = \mathrm{Var}\left(|y_k^* - f(\boldsymbol{x}_k)|\right)$$
$$= \mathrm{Var}\left(|f(\boldsymbol{x}^*) - f(\boldsymbol{x}_k)|\right)$$
$$= \underbrace{\mathbb{E}_{\theta,\theta^*}\left[\mathrm{Var}_{\boldsymbol{x}_k,\boldsymbol{x}^*}(|f(\boldsymbol{x}^*) - f(\boldsymbol{x}_k)| \mid \theta, \theta^*)\right]}_{T_1} + \underbrace{\mathrm{Var}_{\theta,\theta^*}(\mathbb{E}_{\boldsymbol{x}_k,\boldsymbol{x}^*}[|f(\boldsymbol{x}^*) - f(\boldsymbol{x}_k)| \mid \theta, \theta^*])}_{T_2}.$$

In particular, the first term T_1 corresponds to the aleatoric component and the second term T_2 corresponds to the episdemic component. We then proceed to bound the above two terms separately.

Step 1: bound T_1. Under Assumption 2,

$$\mathrm{Var}_{\boldsymbol{x}_k,\boldsymbol{x}^*}(|f(\boldsymbol{x}^*)-f(\boldsymbol{x}_k)|\mid\theta,\theta^*)\leq L^2\mathrm{Var}_{\boldsymbol{x}_k,\boldsymbol{x}^*}(\|\boldsymbol{x}^*-\boldsymbol{x}_k\|\mid\mid\theta,\theta^*).$$

Under Assumption 3 and by Lemma B.2,

$$T_1\leq L^2\mathbb{E}_{\theta,\theta^*}[\mathrm{Var}_{\boldsymbol{x}_k,\boldsymbol{x}^*}(\|\boldsymbol{x}^*-\boldsymbol{x}_k\|\mid\mid\theta,\theta^*)]\leq c_3L^2d\sigma^2. \tag{28}$$

Step 2: bound T_2. Under Assumption 2,

$$T_2\leq L^2\mathrm{Var}_{\theta,\theta^*}(\mathbb{E}_{\boldsymbol{x}_k,\boldsymbol{x}^*}[\|\boldsymbol{x}^*-\boldsymbol{x}_k\|\mid\mid\theta,\theta^*])$$

By Lemma B.1,

$$\begin{aligned}\mathrm{Var}_{\theta,\theta^*}(\mathbb{E}_{\boldsymbol{x}_k,\boldsymbol{x}^*}[|f(\boldsymbol{x}^*)-f(\boldsymbol{x}_k)|\mid\theta,\theta^*])&\leq\mathrm{Var}_{\theta,\theta^*}\left(\mathbb{E}_{\theta,\theta^*}\left[8\sqrt{d}\sigma+\|\mathbb{E}_{\boldsymbol{x}^*}[\boldsymbol{x}^*]-\mathbb{E}_{\boldsymbol{x}_k}[\boldsymbol{x}_k]\|\right]\right)\\&\leq\mathrm{Var}_{\theta,\theta^*}\left(\|\mathbb{E}_{\boldsymbol{x}^*}[\boldsymbol{x}^*]-\mathbb{E}_{\boldsymbol{x}_k}[\boldsymbol{x}_k]\|\right).\end{aligned}$$

Then by property of variance, we have

$$\mathrm{Var}_{\theta,\theta^*}(\|\mathbb{E}_{\boldsymbol{x}^*}[\boldsymbol{x}^*]-\mathbb{E}_{\boldsymbol{x}_k}[\boldsymbol{x}_k]\|)=\mathbb{E}_{\theta,\theta^*}\left[\|\mathbb{E}_{\boldsymbol{x}^*}[\boldsymbol{x}^*]-\mathbb{E}_{\boldsymbol{x}_k}[\boldsymbol{x}_k]\|^2\right]-\left(\mathbb{E}_{\theta,\theta^*}\left[\|\mathbb{E}_{\boldsymbol{x}^*}[\boldsymbol{x}^*]-\mathbb{E}_{\boldsymbol{x}_k}[\boldsymbol{x}_k]\|\right]\right)^2.$$

From the proof of Lemma B.2, we have

$$\begin{aligned}&\mathbb{E}_{\theta,\theta^*}\left[\|\mathbb{E}_{\boldsymbol{x}^*}[\boldsymbol{x}^*|\theta^*]-\mathbb{E}_{\boldsymbol{x}_k}[\boldsymbol{x}_k|\theta]\|^2\right]\\&=\mathbb{E}_{\theta^*}[\mathbb{E}_{\boldsymbol{x}^*}[\|\boldsymbol{x}^*\|^2|\theta^*]]+\mathbb{E}_{\theta}[\mathbb{E}_{\boldsymbol{x}_k}[\|\boldsymbol{x}_k\|^2|\theta]]-2\mathbb{E}_{\theta,\theta^*}[\mathbb{E}_{\boldsymbol{x}_k}[(\boldsymbol{x}_k|\theta)]^{\mathrm{T}}\mathbb{E}_{\boldsymbol{x}^*}[\boldsymbol{x}^*|\theta^*]]\\&=2(\mathbb{E}_{\theta}[\mathbb{E}_{\boldsymbol{x}_k}[\|\boldsymbol{x}_k\|^2|\theta]]-\mathbb{E}_{\theta,\theta^*}[\mathbb{E}_{\boldsymbol{x}_k}[(\boldsymbol{x}_k|\theta)]^{\mathrm{T}}\mathbb{E}_{\boldsymbol{x}^*}[\boldsymbol{x}^*|\theta^*]])\\&=2(\mathbb{E}_{\theta}[\mathbb{E}_{\boldsymbol{x}_k}[\|\boldsymbol{x}_k\|^2|\theta]]-\|\mathbb{E}_{\theta}[\mathbb{E}_{\boldsymbol{x}_k}[\boldsymbol{x}_k|\theta]]\|^2)\\&=2\mathrm{Var}_{\theta}(\mathbb{E}_{\boldsymbol{x}_k}[\|\boldsymbol{x}_k\|]),\end{aligned}$$

where the third equality is by the law of total expectation and the fact that $\mathbb{E}_{\theta}[\mathbb{E}_{\boldsymbol{x}_k}[\boldsymbol{x}_k|\theta]]=\mathbb{E}_{\theta^*}[\mathbb{E}_{\boldsymbol{x}^*}[\boldsymbol{x}^*|\theta^*]]$. Combining the above results, we have

$$T_2\leq L^2\mathrm{Var}_{\theta,\theta^*}(\|\mathbb{E}_{\boldsymbol{x}^*}[\boldsymbol{x}^*]-\mathbb{E}_{\boldsymbol{x}_k}[\boldsymbol{x}_k]\|)\leq 2L^2\mathrm{Var}_{\theta}(\mathbb{E}_{\boldsymbol{x}_k}[\|\boldsymbol{x}_k\|]). \tag{29}$$

Combining Equation (28) and Equation (29) completes the proof:

$$\mathrm{Var}\left(\Delta(p_\theta,y_k^*)\right)\leq c_3L^2d\sigma^2+2L^2\mathrm{Var}_{\theta}(\mathbb{E}_{\boldsymbol{x}}[\|\boldsymbol{x}_k\|]).$$

□

B.1 Supporting Lemmas

Lemma B.3 (Wainwright (2019)). *Let $\boldsymbol{x}\in\mathbb{R}^d$ be a σ-subGaussian ramdom vector, then*

$$\mathbb{E}[\|\boldsymbol{x}-\mathbb{E}[\boldsymbol{x}]\|]\leq 4\sigma\sqrt{d}. \tag{30}$$

Lemma B.4. *Let $\boldsymbol{x}\in\mathbb{R}^d$ be a σ-subGaussian ramdom vector, then its variance satisfies:*

$$\mathit{Var}[\boldsymbol{x}]\leq Cd\sigma^2, \tag{31}$$

where C is some positive constant.

Proof of lemma B.4. By definition of sub-Gaussian vector, for any direction $\boldsymbol{u}\in\mathbb{R}^d$ with $\|\boldsymbol{u}\|=1$,

$$\mathbb{E}\left[\exp(\lambda\boldsymbol{u}^{\mathrm{T}}(\boldsymbol{x}-\mathbb{E}[\boldsymbol{x}]))\right]\leq\exp\left(\frac{\lambda^2\sigma^2}{2}\right),\quad\forall\lambda\in\mathbb{R}.$$

This implies that the second moment in any direction $\boldsymbol{u}$ satisfies:

$$\mathbb{E}\left[\boldsymbol{u}^{\mathrm{T}}((\boldsymbol{x}-\mathbb{E}[\boldsymbol{x}])(\boldsymbol{x}-\mathbb{E}[\boldsymbol{x}])^{\mathrm{T}})\right]\leq\sigma^2.$$

Therefore, the maximum eigenvalue of the covariance matrix is upper-bounded by $C\sigma^2$, where C is some positive constant.

$$\mathrm{Var}[\boldsymbol{x}]=\mathrm{tr}\left(\mathbb{E}\left[(\boldsymbol{x}-\mathbb{E}[\boldsymbol{x}])(\boldsymbol{x}-\mathbb{E}[\boldsymbol{x}])^{\mathrm{T}}\right]\right)\leq Cd\sigma^2.$$

□

Lemma B.5. *In each iteration $k \in [K]$, let $\mathcal{D}$ be the collected dataset, θ and θ^* are parameters independently drawn from posterior $p(\theta|\mathcal{D})$, $\boldsymbol{x}_k \sim p_\theta(\cdot|y_k^*, \mathcal{D})$ and $\boldsymbol{x}^* \sim p_{\theta^*}(\cdot|y_k^*, \mathcal{D})$. For any measurable function f, and $\sigma(\mathcal{D})$-measurable random variable $\boldsymbol{x_k}$,*

$$\mathbb{E}\left[f(\boldsymbol{x}_k)\right] = \mathbb{E}\left[f(\boldsymbol{x}^*)\right].$$

Proof of Lemma B.5. Since the black-box function f is measurable, and by the nature of Algorithm 1, in each iteration k, the generated sample $\boldsymbol{x_k}$, the target function value y_k^*, the predictive distribution $p_\theta(\cdot|y_k^*, \mathcal{D})$, the posterior distribution $p(\theta \mid \mathcal{D})$ are $\sigma(\mathcal{D})$-measurable at iteration k, the only randomness in $f(\boldsymbol{x})$ comes from the random sampling in the algorithm. Thus, condition on the training data $\mathcal{D}$ and target value y_k^*, by tower rule,

$$\begin{aligned}\mathbb{E}\left[f(\boldsymbol{x}_k)\right] = \mathbb{E}\left[\mathbb{E}\left[f(\boldsymbol{x}_k)|\theta\right]\right] &= \int_\theta \int_{\boldsymbol{x}_k} f(\boldsymbol{x}_k) p_\theta(\boldsymbol{x}_k|y_k^*, \mathcal{D}) p(\theta|\mathcal{D}) \,\mathrm{d}\boldsymbol{x}_k \,\mathrm{d}\theta \\ &= \int_\theta \int_{\boldsymbol{x}_k} f(\boldsymbol{x}_k) p_\theta(\boldsymbol{x}_k|y_k^*, \mathcal{D}) \,\mathrm{d}\boldsymbol{x}_k \; p(\theta|\mathcal{D}) \,\mathrm{d}\theta.\end{aligned}$$

Note that both the true parameter θ^* and the chosen parameter θ are drawn from the same posterior distribution $p(\theta \mid \mathcal{D})$, we have

$$\int_\theta \int_{\boldsymbol{x}} f(\boldsymbol{x}) p_\theta(\boldsymbol{x}|y_k^*, \mathcal{D}) \,\mathrm{d}\boldsymbol{x} \; p(\theta|\mathcal{D}) \,\mathrm{d}\theta = \int_{\theta^*} \int_{\boldsymbol{x}} f(\boldsymbol{x}) p_{\theta^*}(\boldsymbol{x}|y_k^*, \mathcal{D}) \,\mathrm{d}\boldsymbol{x} \; p(\theta^*|\mathcal{D}) \,\mathrm{d}\theta^*.$$

As a result, we have

$$\mathbb{E}\left[f(\boldsymbol{x}_k)\right] = \int_{\theta^*} \int_{\boldsymbol{x}^*} f(\boldsymbol{x}^*) p_{\theta^*}(\boldsymbol{x}^*|y_k^*, \mathcal{D}) \,\mathrm{d}\boldsymbol{x}^* p(\theta^*|\mathcal{D}) \,\mathrm{d}\theta^* = \mathbb{E}\left[\mathbb{E}\left[f(\boldsymbol{x}^*)|\theta^*\right]\right] = \mathbb{E}\left[f(\boldsymbol{x}^*)\right].$$

□

Corollary 1. *In each iteration $k \in [K]$, let $\mathcal{D}$ be the collected dataset, θ and θ^* are parameters independently drawn from posterior $p(\theta|\mathcal{D})$, $\boldsymbol{x}_k \sim p_\theta(\cdot|y_k^*, \mathcal{D})$ and $\boldsymbol{x}^* \sim p_{\theta^*}(\cdot|y_k^*, \mathcal{D})$. For any measurable function f, and $\sigma(\mathcal{D})$-measurable random variable $\boldsymbol{x_k}$,*

$$\mathbb{E}\left[\|\boldsymbol{x}_k\|\right] = \mathbb{E}\left[\|\boldsymbol{x}^*\|\right].$$

Proof of Corollary 1. Since the norm function is deterministic and $\sigma(\mathcal{D})$-measurable, the proof directly follows that of Lemma B.5. □

C Optimality of Proposed Acquisition Function

Theorem 4. *Let $\mathcal{Y}$ be the constructed candidate set at each iteration $k \in [K]$ in Algorithm 1. By adopting UaE as the acquisition function to guide the sample generation process in conditional diffusion model, Diff-BBO (Algorithm 1) achieves a near-optimal solution for the online BBO problem defined in Equation (3):*

$$\max_{y_k \in \mathbb{R}} \sum_{k=1}^{K} f(\boldsymbol{x}_k), \quad \boldsymbol{x}_k \sim p_\theta(\cdot \mid y_k, \mathcal{D}), \quad \theta \in \Theta \quad \Rightarrow \max_{y_k \in \mathcal{Y}} \sum_{k=1}^{K} \alpha(y_k, \mathcal{D}).$$

Proof of Theorem 4. Following Theorem 3, we can express the function evaluation as follows,

$$f(\boldsymbol{x}_k) = y_k - (y_k - f(\boldsymbol{x}_k)), \forall k \in [K].$$

The overall objective of the optimization problem defined in Equation (3) can then be further decomposed as

$$\begin{aligned}&\max_{y_k \in \mathbb{R}} \sum_{k=1}^{K} f(\boldsymbol{x}_k), \quad \boldsymbol{x}_k \sim p_\theta(\cdot \mid y_k), \quad \theta \in \Theta \\ \Leftrightarrow &\max_{y_k \in \mathbb{R}} \sum_{k=1}^{K} y_k - (y_k - f(\boldsymbol{x}_k)), \quad \boldsymbol{x}_k \sim p_\theta(\cdot \mid y_k), \quad \theta \in \Theta\end{aligned}$$

$$\Rightarrow \max_{y_k \in \mathcal{Y}} \sum_{k=1}^{K} y_k - \Delta(p_\theta, y_k),$$

where the candidate set $\mathcal{Y}$ is constructed based on the model's predictions and is designed to explore the objective space efficiently. When considering an online maximization problem, adding a positive term would lead to overestimation, because the model would be overly optimistic about $f(\boldsymbol{x}_k)$. Therefore, we should only consider the case where we the uncertainty is being subtracted. By Theorem 3, which shows $\Delta(p_\theta, y_k^*)$ can be effectively upper bounded the epidemic uncertainty, we therefore have

$$\max_{y_k \in \mathbb{R}} \sum_{k=1}^{K} f(\boldsymbol{x}_k), \quad \boldsymbol{x}_k \sim p_\theta(\cdot \mid y_k), \quad \theta \in \Theta \Rightarrow \max_{y_k \in \mathcal{Y}} \sum_{k=1}^{K} y_k - \Delta_{\text{episdemic}}(y_k, \mathcal{D}).$$

Essentially, our chosen acquisition function allows Diff-BBO to maximize the lower bound of the original optimization problem. Penalizing high uncertainty ensures that the model prioritizes more confident predictions (i.e. those with lower epistemic uncertainty), which are more likely to yield higher objective function values. □

D Experiment Details

D.1 Dataset Details.

DesignBench (Trabucco et al., 2022) is a benchmark for real-world black-box optimization tasks. For continuouse tasks, we use Superconductor, D'Kitty Morphology and Ant Morphology benchmarks. For discrete tasks, we utilize TFBind8 and TFBind10 benchmarks. We exclude Hopper due to the domain is known to be buggy, as explained in Appendix C in (Krishnamoorthy et al., 2023). We also exclude NAS due to the significant computational resource requirement. Additionally, we exclude the ChEMBL task because the oracle model exhibits non-trivial discrepancies when queried with the same design.

- **Superconductor (materials optimization).** This task involves searching for materials with high critical temperatures. The dataset comprises 17,014 vectors, each with 86 components that represent the number of atoms of each chemical element in the formula. The provided oracle function is a pre-trained random forest regression model.
- **D'Kitty Morphology (robot morphology optimization).** This task focuses on optimizing the parameters of a D'Kitty robot, including the size, orientation, and location of the limbs, to make it suitable for a specific navigation task. The dataset consists of 10,004 entries with a parameter dimension of 56. It utilizes MuJoCO (Todorov et al., 2012), a robot simulator, as the oracle function.
- **Ant Morphology (robot morphology optimization).** Similar to D'Kitty, this task aims to optimize the parameters of a quadruped robot to maximize its speed. It includes 10,004 data points with a parameter dimension of 60. It also uses MuJoCO as the oracle function.
- **TFBind8 (DNA sequence optimization).** This task seeks to identify the DNA sequence of length eight with the highest binding affinity to the transcription factor SIX6 REF R1. The design space comprises sequences of nucleotides represented as categorical variables. The dataset size is 32,898, with a dimension of 8. The ground truth is used as a direct oracle since the affinity for the entire design space is available.
- **TFBind10 (DNA sequence optimization).** Similar to TFBind8, this task aims to find the DNA sequence of length ten that exhibits the highest binding affinity with transcription factor SIX6 REF R1. The design space consists of all possible nucleotide sequences. The dataset size is 10,000, with a dimension of 10. The ground truth is used as a direct oracle since the affinity for the entire design space is available.

Molecular Discovery. A key problem in drug discovery is the optimization of a compound's activity against a biological target with therapeutic value. Similar to other papers (Eckmann et al., 2022; Jeon & Kim, 2020; Lee et al., 2023; Noh et al., 2022), we attempt to optimize the score from AutoDock4 (Morris et al., 2009), which is a physics-based estimator of binding affinity. The oracle is a feed-forward model as a surrogate to AutoDock4. The surrogate model is trained until convergence on 10,000 compounds randomly sampled from the latent space (using $\mathcal{N}(0, 1)$) and their computed objective values with AutoDock4. We construct our continuous design space by fixing a random protein embedding and randomly sampling 10,000 molecular embedding of dimension 32.

For all the tasks, We sort the offline dataset based on the objective values and select data from the 25% to 50% as the initial training dataset. We use data with lower objective scores to better observe performance differences across each baseline. The overview of all the task statistics is provided in Table 2.

Task	Size	Dimensions	Task Max
TFBind8	32,898	8	1.0
TFBind10	10,000	10	2.128
D'Kitty	10,004	56	340.0
Ant	10,004	60	590.0
Superconductor	17,014	86	185.0
Molecular Discovery	10,000	32	1.0

Table 2: Data Statistics

D.2 Implementation Details.

We train our model on NVIDIA A100 GPU and report the average performance over 3 random runs along with standard deviation for each task. For discrete tasks, we follow the procedure in Krishnamoorthy et al. (2023) where we convert the d-dimensional vector to a $d \times c$ one hot vector regarding c classes. We then approximate logits by interpolating between a uniform distribution and the one hot distribution using a mixing factor of 0.6. We jointly train a conditional and unconditional model with the same model by randomly set the conditioning value to 0 with dropout probability of 0.15.

For each task, we fix the learning rate at 0.001 with batch size of 256. We use 5 ensemble models to estimate the uncertainty for our acquisition function. We set hidden dimensions to 1024 and gamma to 2. We use 10% of the available data at each iteration as validation set during training.

E Impact Statement

Optimization techniques can address various real-world problems, including drug and material design. Our method enhances sample-efficient online black-box optimization, potentially accelerating solutions in these areas. However, caution is needed to prevent misuse, such as optimizing drugs to enhance harmful side effects.