Feedback Efficient Online Fine-Tuning of Diffusion Models

Masatoshi Uehara [*1] **Yulai Zhao** [*2] **Kevin Black** [3] **Ehsan Hajiramezanali** [1] **Gabriele Scalia** [1]
Nathaniel Lee Diamant [1] **Alex M Tseng** [1] **Sergey Levine** [†3] **Tommaso Biancalani** [†1]

Abstract

Diffusion models excel at modeling complex data distributions, including those of images, proteins, and small molecules. However, in many cases, our goal is to model parts of the distribution that maximize certain properties: for example, we may want to generate images with high aesthetic quality, or molecules with high bioactivity. It is natural to frame this as a reinforcement learning (RL) problem, in which the objective is to fine-tune a diffusion model to maximize a reward function that corresponds to some property. Even with access to online queries of the ground-truth reward function, efficiently discovering high-reward samples can be challenging: they might have a low probability in the initial distribution, and there might be many infeasible samples that do not even have a well-defined reward (e.g., unnatural images or physically impossible molecules). In this work, we propose a novel reinforcement learning procedure that efficiently explores on the manifold of feasible samples. We present a theoretical analysis providing a regret guarantee, as well as empirical validation across three domains: images, biological sequences, and molecules.

1. Introduction

Diffusion models belong to the family of deep generative models that reverse a diffusion process to generate data from noise. These diffusion models are highly adept at capturing complex spaces, such as the manifold of natural images, biological structures (e.g., proteins and DNA sequences), or chemicals (e.g., small molecules) (Watson et al., 2023; Jing et al., 2022; Alamdari et al., 2023; Zhang et al., 2024; Igashov et al., 2022). However, in many cases, our goal is to focus on parts of the distribution that maximize certain properties. For example, in chemistry, one might train a diffusion model on a wide range of possible molecules, but applications of such models in drug discovery might need to steer the generation toward samples with high bioactivity. Similarly, in image generation, diffusion models are trained on large image datasets scraped from the internet, but practical applications often desire images with high aesthetic quality. This challenge be framed as a reinforcement learning (RL) problem, where the objective is to fine-tune a diffusion model to maximize a reward function that corresponds to the desirable property.

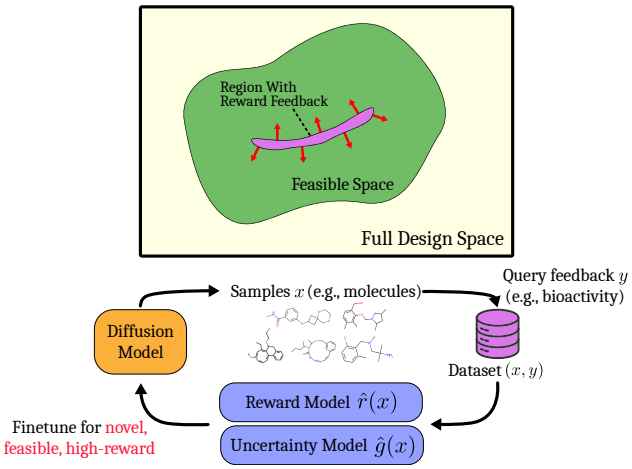

Figure 1. We consider a scenario where we have a pre-trained diffusion model capturing a feasible region embedded as an intricate manifold (green) in a high-dimensional full design space (yellow). We aim to fine-tune the model to get feasible samples with high rewards and novelty. The purple area corresponds to the region where we have reward feedback, which we aim to expand with efficient online queries to the reward function.

One significant obstacle in many applications is the cost of acquiring feedback from the ground-truth reward function. For instance, in biology or chemistry, evaluating ground truth rewards requires time-consuming wet lab experiments. In image generation, attributes such as aesthetic quality are subjective and require human judgment to establish ground truth scores. While several recent works have pro-

[*]Equal contribution: ueharam.masatoshi@gene.com, yulaiz@princeton.edu [1]Genentech [2]Princeton University [3]University of California, Berkeley. Correspondence to: Sergey Levine <sergey.levine@berkeley.edu>, Tommaso Biancalani <biancalt@gene.com>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

posed methods for RL-based fine-tuning of diffusion models (Black et al., 2023; Fan et al., 2023; Clark et al., 2023; Prabhudesai et al., 2023; Uehara et al., 2024), none directly address the issue of feedback efficiency in an online setting. This motivates us to develop an online fine-tuning method that minimizes the number of reward queries.

Achieving this goal demands efficient exploration. But, because we are dealing with high-dimensional spaces, this entails more than just exploring new regions. It also involves adhering to the structural constraints of the problem. For example, in biology, chemistry, and image generation, valid points such as physically feasible molecules or proteins typically sit on a lower-dimensional manifold (the "feasible space") embedded within a much larger full design space. Thus, an effective feedback-efficient fine-tuning method should explore the space without leaving this feasible area, as this would result in wasteful invalid queries.

Hence, we propose a feedback-efficient iterative fine-tuning approach for diffusion models that intelligently explores the feasible space, as illustrated in Figure 1. Here, in each iteration, we acquire new samples from the current diffusion model, query the reward function, and integrate the result into the dataset. Then, using this augmented dataset with reward feedback, we train the reward function and its uncertainty model, which assigns higher values to areas not well-covered by the current dataset (i.e., more novel regions). We then update the diffusion model using the learned reward function, enhanced by its uncertainty model and a Kullback-Leibler (KL) divergence term relative to the current diffusion model, but without querying new feedback. This fine-tuned diffusion model then explores regions in the feasible space with high rewards and high novelty in the next iteration.

Our main contribution is to propose a provably feedback-efficient method for the RL-based online fine-tuning of diffusion models, compared to existing works focused on improving computational efficiency. Our conceptual and technical innovation lies in introducing a novel online method tailored to diffusion models by (a) interleaving reward learning and diffusion model updates and (b) integrating an uncertainty model and KL regularization to facilitate exploration while containing it to the feasible space. Furthermore, we show the feedback efficiency of our method by providing a regret guarantee, and experimentally validate our proposed method on a range of domains, including images, protein sequences.

2. Related Works

Here, we offer a comprehensive review of related literature.

Fine-tuning diffusion models. Many prior works have endeavored to fine-tune diffusion models by optimizing reward functions through supervised learning (Lee et al., 2023; Wu et al., 2023), policy gradient, (Black et al., 2023; Fan et al., 2023), or control-based (i.e., direct backpropagation) techniques (Clark et al., 2023; Xu et al., 2023; Prabhudesai et al., 2023). Most importantly, these existing work assumes a static reward model; the rewards are either treated as ground truth, or if not, no allowance is made for online queries to ground-truth feedback. In contrast, we consider an online setting in which additional queries to the ground-truth reward function are allowed; we then tackle the problems of exploration and feedback efficiency by explicitly including a reward modeling component that is interleaved with diffusion model updates. Our experiments show improved feedback efficiency over prior methods across various domains, including images, biological sequences, and molecules, whereas prior work has only considered images. Finally, our paper includes key theoretical results (Theorem 1) absent from prior work.

Online learning with generative models. Feedback-efficient online learning has been discussed in several domains, such as NLP and CV (Brantley et al., 2021; Zhang et al., 2022). For example, in LLMs, fine-tuning methods that incorporate human feedback (RLHF), such as preferences, have gained widespread popularity (Touvron et al., 2023; Ouyang et al., 2022). Based on that, Dong et al. (2023) introduces a general online learning approach for generative models. However, their approach doesn't appear to be tailored specifically for diffusion models, which sets it apart from our work. Notably, their fine-tuning step relies on supervised fine-tuning, unlike our work. From a related perspective, recent studies (Yang et al., 2023; Wallace et al., 2023) have explored fine-tuning techniques of diffusion models using human feedback. However, their emphasis differs significantly in that they aim to leverage preference-based feedback by directly optimizing preferences without constructing a reward model following the idea of Direct Preference Optimization (Rafailov et al., 2024). In contrast, our focus is on proposing a provably feedback-efficient method while still utilizing non-preference-based feedback.

Guidance. Dhariwal and Nichol (2021) introduced classifier-based guidance, an inference-time technique for steering diffusion samples towards a particular class. More generally, guidance uses an auxiliary differentiable objective (e.g., a neural network) to steer diffusion samples towards a desired property (Graikos et al., 2022; Bansal et al., 2023). However, we would expect this approach to work best when the guidance network is trained on high-quality data with reward feedback that comprehensively covers the sample space. Our emphasis lies in scenarios where we lack such high-quality data and must gather more in an online fashion. Indeed, we show in our experiments that our method outperforms a guidance baseline that merely uses the gradients of

the reward model to steer the pre-trained diffusion model toward desirable samples.

Bandits and black-box optimization. Adaptive data collection has been a topic of study in the bandit literature, and our algorithm draws inspiration from this body of work (Lattimore and Szepesvári, 2020). However, in traditional literature, the action space is typically small. Our challenge lies in devising a practical algorithm capable of managing a complicated and large action space. We do this by integrating a pre-trained diffusion model. Although there exists literature on bandits with continuous actions using classical nonparametric methods (Krishnamurthy et al., 2020; Frazier, 2018; Balandat et al., 2020), these approaches struggle to effectively capture the complexities of image, biological, or chemical spaces. Some progress in this regard has been made through the use of variational autoencoders (VAEs) (Kingma and Welling, 2013) as shown in Notin et al. (2021); Gómez-Bombarelli et al. (2018). However, our focus is on how to incorporate pre-trained diffusion models.

Note that although there are several exceptions discussing black-box optimization with diffusion models (Wang et al., 2022; Krishnamoorthy et al., 2023), their fine-tuning methods rely on guidance and are more tailored to the offline setting. In contrast, we employ the RL-based fine-tuning method, and our setting is online. We will also compare our approach with guided methods in Section 7.

3. Preliminaries

We consider an online bandit setting with a pre-trained diffusion model. Before explaining our setting, we provide an overview of our relation to bandits and diffusion models.

Bandits. Let's consider a design space denoted as $\mathcal{X}$. Within $\mathcal{X}$, each element x is associated with a corresponding reward $r(x)$. Here, $r : \mathcal{X} \to [0, 1]$ represents an unknown reward function. A common primary objective is to strategically collect data and learn this reward function so that we can discover high-quality samples that have high $r(x)$. For example, in biology, our goal may be to discover protein sequences with high bioactivity.

As we lack knowledge of the reward function $r(\cdot)$, our task is to learn it by collecting feedback gradually. We focus on situations where we can access noisy feedback $y = r(x) + \epsilon$ for a given sample x. In real-world contexts, such as chemistry or biology, this feedback frequently comprises data obtained from wet lab experiments. Given the expense associated with obtaining feedback, our objective is to efficiently identify high-quality samples while minimizing the number of feedback queries. This problem setting is commonly referred to as a bandit or black-box optimization (Lattimore and Szepesvári, 2020).

Diffusion Models. A diffusion model is described by the following equation:

$$dx_t = f(t, x_t)dt + \sigma(t)dw_t,\ x_0 \sim \nu, \tag{1}$$

where $f : [0, T] \times \mathcal{X} \to \mathcal{X}^d$ is a drift coefficient, $\sigma : [0, T] \to \mathbb{R}_{>0}$ is a diffusion coefficient associated with a d-dimensional Brownian motion w_t, and $\nu \in \Delta(\mathcal{X})$ is an initial distribution. It's worth noting that some papers use the reverse notation. When training diffusion models, the goal is to learn $f(t, x_t)$ from the data so that the generated distribution following the SDE aligns with the data distribution through score matching (Song et al., 2020; Ho et al., 2020) or flow matching (Lipman et al., 2023; Shi et al., 2023; Tong et al., 2023; Somnath et al., 2023; Albergo et al., 2023; Liu et al., 2023; 2022).

In our study, we explore a scenario where we have a pre-trained diffusion model described by the following SDE:

$$dx_t = f^{\mathrm{pre}}(t, x_t)dt + \sigma(t)dw_t,\ x_0 \sim \nu^{\mathrm{pre}}, \tag{2}$$

where f^{pre} is a drift coefficient and ν^{pre} is an initial distribution for the pre-trained model. We denote the generated distribution (i.e., marginal distribution at T following (2)) by p^{pre}.

Notation. We often use $[K]$ to denote $[1, \cdots, K]$. For $q, p \in \Delta(\mathcal{X})$, the notation $q \propto p$ means q is equal to p up to normalizing constants. We denote the KL divergence between $p, q \in \Delta(\mathcal{X})$ by $\mathrm{KL}(p\|q)$. We often consider a measure $\mathbb{P}$ induced by an SDE on $\mathcal{C} := C([0, T], \mathcal{X})$ where $C([0, T], \mathcal{X})$ is the whole set of continuous functions on mapping from $[0, T]$ to $\mathcal{X}$. When this SDE is associated with a drift term f and an initial distribution ν as in (1), we denote the measure by $\mathbb{P}^{u,\nu}$. The notation $\mathbb{E}_{\mathbb{P}^{u,\nu}}[f(x_{0:T})]$ means that the expectation is taken for $f(\cdot)$ with respect to $\mathbb{P}^{u,\nu}$. We denote the marginal distribution of $\mathbb{P}^{u,\nu}$ over $\mathcal{X}$ at time t by $\mathbb{P}^{u,\nu}_t$. We also denote the distribution of the process conditioned on an initial and terminal point x_0, x_T by $\mathbb{P}_{\cdot|0,T}(\cdot|x_0, x_T)$ (we similarly define $\mathbb{P}_{\cdot|T}(\cdot|x_T)$). With a slight abuse of notation, we exchangeably use distributions and densities.

In this work, we defer all proofs to Appendix B.

4. Problem Statement: Efficient RL Fine-Tuning of Diffusion Models

In this section, we state our problem setting: online bandit fine-tuning of pre-trained diffusion models.

In contrast to standard bandit settings, which typically operate on a small action space $\mathcal{X}$, our primary focus is on addressing the complexities arising from an exceedingly vast design space $\mathcal{X}$. For instance, when our objective involves generating molecules with graph representation, the

Table 1. Examples of original space and feasible space.

	Original space ($\mathcal{X}$)	Feasible space ($\mathcal{X}_{\text{pre}}$)
Images	3-dimensional tensors	Natural images
Bio	Sequence ($\lvert 20\rvert^B, \lvert 4\rvert^B$)	Natural proteins/DNA
Chem	Graphs	Natural molecules

cardinality of $|\mathcal{X}|$ is huge. While this unprocessed design space exhibits immense scale, the actual feasible and meaningful design space often resides within an intricate but potentially low-dimensional manifold embedded in $\mathcal{X}$. In biology, this space is frequently referred to as the biological space. Similarly, in chemistry, it is commonly referred to as the chemical space. Notably, recent advances have introduced promising diffusion models aimed at capturing the intricacies of these biological or chemical spaces (Vignac et al., 2023; Tseng et al., 2023; Avdeyev et al., 2023). We denote such a feasible space by $\mathcal{X}_{\text{pre}}$, as in Table 1.

Now, let's formalize our problem. We consider feedback-efficient online fine-tuning of diffusion models. Specifically, we work on a bandit setting where we do not have any data with feedback initially, but we have a pre-trained diffusion model trained on a distribution $p^{\text{pre}}(\cdot)$, whose support is given by $\mathcal{X}_{\text{pre}}$. We aim to fine-tune this diffusion model to produce a new model $p \in \Delta(\mathcal{X})$ that maximizes $\mathbb{E}_{x\sim p(x)}[r(x)]$, where $r(\cdot)$ is 0 outside the support of $\mathcal{X}_{\text{pre}}$ ($\mathcal{X}_{\text{pre}} := \{x \in \mathcal{X} : p^{\text{pre}}(x) > 0\}$) because a data not in $\mathcal{X}_{\text{pre}}$ is invalid. By leveraging the fine-tuned model, we want to efficiently explore the vast design space $\mathcal{X}$ (i.e., minimizing the number of queries to the true reward function) while avoiding generation of invalid instances lying outside of $\mathcal{X}_{\text{pre}}$.

5. Algorithm

In this section, we present our novel framework **SEIKO** for fine-tuning diffusion models. The core of our method consists of two key components: (a) effectively preserving the information from the pre-trained diffusion model through KL regularization, enabling exploration within the feasible space $\mathcal{X}_{\text{pre}}$, (b) introducing an optimistic bonus term to facilitate the exploration of novel regions of $\mathcal{X}_{\text{pre}}$.

Our algorithm follows an iterative approach. Each iteration comprises three key steps: (i) feedback collection phase, (ii) reward model update with feedback, and (iii) diffusion model update. Here, we decouple the feedback collection phase (i) and diffusion model update (iii) so that we do not query new feedback to the true reward when updating the diffusion model in step (iii). Algorithm 1 provides a comprehensive outline of **SEIKO** (Optimistic Finetuning of Diffusion models with KL constraint). We will elaborate on each component in the subsequent sections.

Algorithm 1 SEIKO (Optimi**S**tic fin**E**-tuning of d**I**ffusion with **K**L c**O**nstraint)

1: **Require**: Parameter $\alpha, \{\beta_i\} \in \mathbb{R}^+$, a pre-trained diffusion model described by $f^{\text{pre}} : [0, T] \times \mathcal{X} \to \mathcal{X}$, an initial distribution $\nu^{\text{pre}} : \mathcal{X} \to \Delta(\mathcal{X})$
2: Initialize $f^{(0)} = f^{\text{pre}}, \nu^{(0)} = \nu^{\text{pre}}$
3: **for** i in $[1, \cdots, K]$ **do**
4: Generate a new sample $x^{(i)} \sim p^{(i-1)}(x)$ following

$$dx_t = f^{(i-1)}(t, x_t)dt + \sigma(t)dw_t,\ x_0 \sim \nu^{(i-1)}, \tag{3}$$

from 0 to T, and get a feedback $y^{(i)} = r(x^{(i)}) + \epsilon$. (Note $p^{(0)} = p^{\text{pre}}$.)
5: Construct a new dataset: $\mathcal{D}^{(i)} = \mathcal{D}^{(i-1)} \cup (x^{(i)}, y^{(i)})$
6: Train a reward model $\hat{r}^{(i)}(x)$ and uncertainty oracle $\hat{g}^{(i)}(x)$ using a dataset $\mathcal{D}^{(i)}$
7: Update a diffusion model by solving the control problem:

$$\begin{aligned} f^{(i)}, \nu^{(i)} = \underset{f:[0,T]\times\mathcal{X}\to\mathcal{X}, \nu\in\Delta(\mathcal{X})}{\operatorname{argmax}} & \underbrace{\mathbb{E}_{\mathbb{P}f,\nu}\left[(\hat{r}^{(i)} + \hat{g}^{(i)})(x_T)\right]}_{\text{(B) Optimistic reward}} \\ & - \alpha \underbrace{\mathbb{E}_{\mathbb{P}f,\nu}\left[\log\left(\frac{\nu(x_0)}{\nu^{\text{pre}}(x_0)}\right) + \int_{t=0}^{T} \frac{\|g^{(0)}(t, x_t)\|^2}{2\sigma^2(t)} dt\right]}_{\text{(A1) KL Regularization relative to a diffusion at iteration 0}} \\ & - \beta_{i-1} \underbrace{\mathbb{E}_{\mathbb{P}f,\nu}\left[\log\left(\frac{\nu(x_0)}{\nu^{(i-1)}(x_0)}\right) + \int_{t=0}^{T} \frac{\|g^{(i-1)}(t, x_t)\|^2}{2\sigma^2(t)} dt\right]}_{\text{(A2) KL Regularization relative to a diffusion at iteration } i-1} \end{aligned} \tag{4}$$

where $g^{(i-1)} := f^{(i-1)} - f$, $g^{(0)} := f^{(0)} - f$ and the expectation $\mathbb{E}_{\mathbb{P}f,\nu}[\cdot]$ is taken with respect to the distribution induced by the SDE associated with a drift f and an initial distribution ν in (1). Refer to Appendix A regarding algorithms to solve (4).
8: **end for**
9: **Output**: $p^{(1)}, \cdots, p^{(K)}$.

5.1. Data Collection Phase (Line 4)

We consider an iterative procedure. Hence, at this iteration $i \in [K]$, we have a fine-tuned diffusion model (when $i = 1$, this is just a pre-trained diffusion model), that is designed to explore designs with high rewards and high novelty in Subsection 5.3.

Using this diffusion model, we obtain new samples in each iteration and query feedback for these samples. To elaborate, in each iteration, we maintain a drift term $f^{(i-1)}$ and an initial distribution $\nu^{(i-1)}$. Then, following the SDE associated with $f^{(i-1)}$ and $\nu^{(i-1)}$ in (3), we construct a data-collection distribution $p^{(i-1)} \in \Delta(\mathcal{X})$ (i.e. a marginal distribution at T). After getting a sample $x^{(i)}$, we obtain its corresponding feedback $y^{(i)}$. Then, we aggregate this new pair $\{x^{(i)}, y^{(i)}\}$ into the existing dataset $\mathcal{D}^{(i-1)}$ in Line 5.

Remark 1 (Batch online setting)**.** *To simplify the notation, we present an algorithm for the scenario where one sample is generated during each online epoch. In practice, multiple samples may be collected. In such cases, our algorithm remains applicable, and the theoretical guarantee holds with minor adjustments (Gao et al., 2019).*

5.2. Reward Model Update (Line 6)

We learn a reward model $\hat{r}^{(i)} : \mathcal{X} \to \mathbb{R}$ from the dataset $\mathcal{D}^{(i)}$. To construct $\hat{r}$, a typical procedure is to solve the (regularized) empirical risk minimization (ERM) problem:

$$\hat{r}^{(i)}(\cdot) = \operatorname*{argmin}_{r \in \mathcal{F}} \sum_{(x,y)\sim\mathcal{D}^{(i)}} \{r(x) - y\}^2 + \|r\|_{\mathcal{F}}, \quad (5)$$

where $\mathcal{F}$ is a hypothesis class such that $\mathcal{F} \subset [\mathcal{X} \to \mathbb{R}]$ and $\|\cdot\|_{\mathcal{F}}$ is a certain norm to define a regularizer.

We also train an uncertainty oracle, denoted as $\hat{g}^{(i)} : \mathcal{X} \to [0, 1]$, using the data $\mathcal{D}^{(i)}$. The purpose of this uncertainty oracle $\hat{g}^{(i)}$ is to assign higher values when $\hat{r}^{(i)}$ exhibits greater uncertainty. We leverage this uncertainty oracle $\hat{g}^{(i)}$ to facilitate exploration beyond the current dataset when we update the diffusion model, as we will see in Section 5.3. This can be formally expressed as follows.

Definition 1 (Uncertainty oracle)**.** *With probability* $1 - \delta$*,*

$$x \in \mathcal{X}_{\text{pre}};\ |\hat{r}^{(i)}(x) - r(x)| \le \hat{g}^{(i)}(x).$$

Note that the above only needs to hold within $\mathcal{X}_{\text{pre}}$*.*

Provided below are some illustrative examples of such $\hat{g}^{(i)}$.

Example 1 (Linear models)**.** *When we use a linear model* $\{x \mapsto \theta^\top \phi(x)\}$ *with feature vector* $\phi : \mathcal{X} \to \mathbb{R}^d$ *for* $\mathcal{F}$*, we use* $(\hat{r}^{(i)}, \hat{g}^{(i)})$ *as follows:*

$$\hat{r}^{(i)}(\cdot) = \phi^\top(\cdot)(\Sigma_i + \lambda I)^{-1} \sum_{j=1}^{i} \phi(x^{(j)}) y^{(j)},$$
$$\hat{g}^{(i)}(\cdot) = C_1(\delta) \min\left(1, \sqrt{\phi(\cdot)^\top (\Sigma_i + \lambda I)^{-1} \phi(\cdot)}\right),$$

where $\Sigma_i = \sum_{j=1}^{i} \phi(x^{(j)})\phi(x^{(j)})^\top$ *,* $C_1(\delta), \lambda \in \mathbb{R}^+$*. This satisfies Definition 1 with a proper choice of* $C_1(\delta)$ *as explained in Agarwal et al. (2019) and Appendix B.1. This can also be extended to cases when using RKHS (a.k.a. Gaussian processes) as in Srinivas et al. (2009); Garnelo et al. (2018); Valko et al. (2013); Chang et al. (2021).*

A practical question is how to choose a feature vector ϕ*. In practice, we recommend using the last layer of neural networks as* ϕ *(Zhang et al., 2022; Qiu et al., 2022).*

Example 2 (Neural networks)**.** *When we use neural networks for* $\mathcal{F}$*, a typical construction of* $\hat{g}^{(i)}$ *is based on statistical bootstrap (Efron, 1992). Many practical bootstrap methods with neural networks have been proposed in Osband et al. (2016); Chua et al. (2018), and its theory has been analyzed (Kveton et al., 2019). Specifically, in a typical procedure, given a dataset* $\mathcal{D}$*, we resample with a replacement for* M *times and get* $\mathcal{D}_1, \cdots, \mathcal{D}_M$*. We train an individual model* $\{\hat{r}^{(i,j)}\}_{j=1}^{M}$ *with each bootstrapped dataset. Then, we set* $\hat{r}^{(i)} = \max_{j\in[M]} \hat{r}^{(i,j)}$ *for* $i \in [K]$*.*

5.3. Diffusion Model Update (Line 7)

In this stage, without querying new feedback, we update the diffusion models (i.e., drift coefficient $f^{(i-1)}$). Our objective is to fine-tune a diffusion model to generate higher-reward samples, exploring regions not covered in the current dataset while using pre-trained diffusion models to avoid deviations from the feasible space.

To achieve this goal, we first introduce an optimistic reward term (i.e., reward with uncertainty oracle) to sample high-reward designs while encouraging exploration. We also include a KL regularization term to prevent substantial divergence between the updated diffusion model and the current one at $i - 1$. This regularization term also plays a role in preserving the information from the pre-trained diffusion model, keeping exploration constrained to the feasible manifold $\mathcal{X}_{\text{pre}}$. The parameters β_i, α govern the magnitude of this regularization term.

Formally, we frame this phase as a control problem in a special version of soft-entropy-regularized MDPs (Neu et al., 2017; Geist et al., 2019), as in Equation (4). In this objective function, we aim to optimize three terms: (B) optimistic reward at the terminal time T, (A1) KL term relative to a pre-trained diffusion model, and (A2) KL term relative to a diffusion model at $i - 1$. Indeed, for terms (A1) and (A2), we use observations:

$$(\text{A1}) = \text{KL}(\mathbb{P}^{f,\nu}(\cdot) \| \mathbb{P}^{f^{\text{pre}},\nu^{\text{pre}}}(\cdot)),$$
$$(\text{A2}) = \text{KL}(\mathbb{P}^{f,\nu}(\cdot) \| \mathbb{P}^{f^{(i-1)},\nu^{(i-1)}}(\cdot)).$$

Importantly, we can show that we are able to obtain a more explicit form for $p^{(i)}(\cdot)$, which is a distribution generated by the fine-tuned diffusion model with $f^{(i)}, \nu^{(i)}$ after solving (4). Indeed, we design a loss function (4) to obtain this form so that we can show our method is provably feedback efficient in Section 6.

Theorem 1 (Explicit form of fine-tuned distributions)**.** *The distribution* $p^{(i)}$ *satisfies*

$$p^{(i)} = \operatorname*{argmax}_{p\in\Delta(\mathcal{X})} \underbrace{\mathbb{E}_{x\sim p}[r(x)]}_{(\text{B}')} - \alpha \underbrace{\text{KL}(p\|p^{(0)})}_{(\text{A1}')} - \beta \underbrace{\text{KL}(p\|p^{(i-1)})}_{(\text{A2}')}. \quad (6)$$

It is equivalent to

$$p^{(i)} \propto \exp\left(\frac{\hat{r}^{(i)}(\cdot) + \hat{g}^{(i)}(\cdot)}{\alpha+\beta}\right) \{p^{(i-1)}(\cdot)\}^{\frac{\beta}{\alpha+\beta}} \{p^{\mathrm{pre}}(\cdot)\}^{\frac{\alpha}{\alpha+\beta}}. \tag{7}$$

Equation (6) states that the $p^{(i)}$ maximizes the function, which comprises three terms: (B'), (A1'), and (A2'), which appear to be similar to (B), (A1), (A2) in (4). Indeed, the term (B') is essentially equivalent to (B) by regarding p as the generated distribution at T by $\mathbb{P}^{f,\nu}$. As for terms (A1') and (A2'), the KL divergences are now defined on the distribution of x_T (over $\mathcal{X}$) unlike (A1), (A2) that are defined on trajectories on $x_{0:T}$ (over $\mathcal{C}$). To bridge this gap, in the proof, we leverage key bridge preserving properties: $(\mathrm{A1}') - (\mathrm{A1}) = 0$ and $(\mathrm{A2}') - (\mathrm{A2}) = 0$. These properties stem from the fact that the posterior distribution over $\mathcal{C}$ on x_T is the same, leading to the conditional KL divergence over $x_{0:T}$ conditioning on x_T being 0.

Equation (7) indicates that $p^{(i)}$ is expressed as the product of three terms: the optimistic reward term, the distribution by the current diffusion model $p^{(i-1)}$, and the pre-trained diffusion model. Through induction, it becomes evident that the support of $p^{(i)}$ always falls within that of the pre-trained model. Consequently, we can avoid invalid designs.

Some astute readers might wonder why we don't directly sample from $p^{(i)}$ in (7). This is because even if we know a drift term f^{pre}, we lack access to the exact form of p^{pre} itself. In our algorithm, without directly trying to estimate p^{pre} and solving the sampling problem, we reduce the sampling problem to a control problem in Equation (4).

Algorithms to solve (4). To solve (4) practically, we can leverage a range of readily available control-based algorithms. Since $\hat{r} + \hat{g}$ is differentiable, we can approximate the expectation over trajectories, $\mathbb{E}_{f,\nu}[\cdot]$, using any approximation method, like Euler-Maruyama, and parameterize f and ν with neural networks. This enables direct optimization of f and ν with stochastic gradient descent, as used in Clark et al. (2023); Prabhudesai et al. (2023); Uehara et al. (2024). For details, refer to Appendix A. We can also use PPO for this purpose Fan et al. (2023); Black et al. (2023).

6. Regret Guarantees

We will demonstrate the provable efficiency of our proposed algorithm in terms of the number of feedback iterations needed to discover a nearly optimal generative model. We quantify this efficiency using the concept of "regret".

To define this regret, first, we define the performance of a generative model $p \in \Delta(\mathcal{X})$ as the following value:

$$J_\alpha(p) := \underbrace{\mathbb{E}_{x\sim p}[r(x)]}_{(\mathrm{B})} - \alpha \underbrace{\mathrm{KL}(p \| p^{\mathrm{pre}})}_{(\mathrm{A1}')}. \tag{8}$$

The first term (B) represents the expected reward of a generative model p. The second term (A1') acts as a penalty when generative models generate samples that deviate from natural data (such as natural molecules or foldable proteins) in the feasible space, as we mention in Section 4. We typically assume that α is very small. This α controls the strength of the penalty term. However, even if α is very small, when p is not covered by p^{pre}, a generative model will incur a substantial penalty, which could be ∞ in extreme cases.

Now, using this value, regret is defined as a quantity that measures the difference between the value of a diffusion model trained with our method and that of a generative model π we aim to compete with (i.e., a high-quality one). While regret is a standard metric in online learning (Lattimore and Szepesvári, 2020), we provide a novel analysis tailored specifically for diffusion models, building upon Theorem 1. For example, while a value is typically defined as $\mathbb{E}_{x\sim\pi}[r(x)]$ in the standard literature on online learning, we consider a new value tailored to our context with pre-trained diffusion models by adding a KL term.

Theorem 2 (Regret guarantee)**.** *With probability $1 - \delta K$, by taking $\beta_i = \alpha(i-1)$, we have the following regret guarantee:* $\forall \pi \in \Delta(\mathcal{X})$;

$$\underbrace{J_\alpha(\pi) - \frac{1}{K}\sum_{k=1}^{K} J_\alpha(p^{(i)})}_{\text{Regret}} \le \underbrace{\frac{2}{K}\sum_{i=1}^{K} \mathbb{E}_{x\sim p^{(i)}}[\hat{g}^{(i)}(x)]}_{\text{Statistical Error}}.$$

This theorem indicates that the performance difference between any generative model $\pi \in \Delta(\mathcal{X})$ and our fine-tuned generative model is effectively controlled by the right-hand side. The right hand side corresponds to statistical error, which typically diminishes as we collect more data, often following rates on the order of $O(1/\sqrt{K})$ under common structural conditions. More specifically, the statistical error term, which depends on the complexity of the function class $\mathcal{F}$ because uncertainty oracles yield higher values as the size of the function class $\mathcal{F}$ expands. While the general form of this error term in Theorem 2 might be hard to interpret, it simplifies significantly for many commonly used function classes. For example, if the true reward is in the set of linear models and we use linear models to represent $\hat{r}$, the statistical error term can be bounded as follows:

Corollary 1 (Statistical error for linear model)**.** *When we use a linear model as in Example 1 such that $\forall x; \|\phi(x)\|_2 \le B$, suppose there exists a function that matches with $r : \mathcal{X} \to \mathbb{R}$ on the feasible space $\mathcal{X}_{\mathrm{pre}}$ (i.e., realizability in $\mathcal{X}_{\mathrm{pre}}$). By taking $\lambda = 1$ and $\delta = 1/H^2$, the expected*

statistical error term in Theorem 2 is upper-bounded by $\tilde{O}(d/\sqrt{K})$ where $\tilde{O}(\cdot)$ hides logarithmic dependence.

Note when we use Gaussian processes and neural networks, we can still replace d by effective dimension (Valko et al., 2013; Zhou et al., 2020).

Remark 2 (α needs to be larger than 0)**.** *In Theorem 2, it is important to set $\alpha > 0$. When $\alpha = 0$, the support of $p^{(i)}$ is not constrained to remain within $\mathcal{X}_{\text{pre}}$. However, $r(\cdot)$ takes 0 outside of $\mathcal{X}_{\text{pre}}$ in our context (See Section 4).*

7. Experiments

Our experiments aim to evaluate our proposed approach in three different domains. We aim to investigate the effectiveness of our approach over existing baselines and show the usefulness of exploration with KL regularization and optimistic exploration.

To start, we provide an overview of these baselines, detail the experimental setups, and specify the evaluation metrics used across all three domains. For comprehensive information on each experiment, including dataset and architecture details and additional results, refer to Appendix D.

Methods to compare. We compare our method with four baseline fine-tuning approaches. The first two are non-adaptive methods that optimize the reward without incorporating adaptive online data collection. The second two are naïve online fine-tuning methods.

- **Non-adaptive (Clark et al., 2023; Prabhudesai et al., 2023)** : We gather M samples (i.e., x) from the pre-trained diffusion model and acquire the corresponding feedback (i.e., y). Subsequently, after training a reward model $\hat{r}$ from this data with feedback, we fine-tune the diffusion model using this static $\hat{r}$ with direct back-propagation.
- **Guidance (a.k.a. classifier guidance) (Dhariwal and Nichol, 2021)** : We collect M samples from the pre-trained diffusion model and receive their feedback y. We train a reward model on this data and use it to guide the sampling process toward high rewards.
- **Online (model-free) PPO:** We run KL-penalized RL finetuning with PPO (Schulman et al., 2017) for M reward queries. This is an improved version of both DPOK (Fan et al., 2023) and DDPO (Black et al., 2023) with a KL penalty applied to the rewards. For details, refer to Appendix D.1.
- **SEIKO**: We collect M_i data at every online epoch in $i \in [K]$ so that the total feedback budget (i.e., $\sum_{i=1}^{K} M_i$) is M as in Remark 1.
  - **Greedy (Baseline)**: Algorithm 1 by setting $\beta = 0$ (no KL term) and $\hat{g}$ to be 0.
  - **Our proposal** with $\beta > 0$ and $\hat{g} \neq 0$.
    - **UCB**: Algorithm 1 with $\hat{g}$ as in Example 1.
    - **Bootstrap**: Algorithm 1 with $\hat{g}$ as in Example 2.

Table 2. Results for fine-tuning diffusion models for protein sequences using the GFP to optimize fluorescence properties. **SEIKO** attains high rewards using a fixed budget of feedback.

	Value ↑	Div ↑
Non-adaptive	3.66 ± 0.03	2.5
Guidance	3.62 ± 0.02	**2.7**
Online PPO	3.63 ± 0.04	2.5
Greedy	3.62 ± 0.00	2.4
UCB (Ours)	3.84 ± 0.00	2.6
Bootstrap (Ours)	$\mathbf{3.87 \pm 0.01}$	2.2

Evaluation. We ran each of the fine-tuning methods 10 times, except for image tasks. In each trial, we trained a generative model $p \in \Delta(\mathbb{R}^d)$. For this p, we measured two metrics: the value, $J_\alpha(p) = \mathbb{E}_{x\sim p}[r(x)] - \alpha \text{KL}(p \| \rho_{\text{pre}})$ in (8) with a very small α (i.e., $\alpha = 10^{-5}$), and the diversity among the generated samples, defined as $\mathbb{E}_{x\sim p, z\sim p}[d(x, z)]$, where $d(x, z)$ quantifies the distance between x and z. . Finally, we reported the mean values of (Value) and (Div) across trials. While our objective is not necessarily focused on just obtaining diverse samples, we included diversity metrics following common conventions in biological tasks.

Setup. We construct a diffusion model by training it on a dataset that includes both high and low-reward samples, which we then employ as our pre-trained diffusion model. Next, we create an oracle r by training it on an extensive dataset with feedback. In our experimental setup, we consider a scenario in which we lack knowledge of r. However, given x, we can get feedback in the form of $y = r(x) + \epsilon$. Our aim is to obtain a high-quality diffusion model that achieves high $\mathbb{E}_{x\sim p}[r(x)]$ by fine-tuning using a fixed feedback budget M.

7.1. Protein Sequences

Our task involves obtaining a protein sequence with desirable properties. As a biological example, we use the GFP dataset (Sarkisyan et al., 2016; Trabucco et al., 2022). In the GFP task, x represents green fluorescent protein sequences, each with a length of 237, and a true $r(x)$ denotes their fluorescence value. We leverage a transformer-based pre-trained diffusion model and oracle using this dataset. To measure distance, we adopt the Levenshtein distance, following Ghari et al. (2023). We set $M_i = 500$ and $K = 4$ ($M = 2000$).

Results. We report results in Table 2. First, it's evident that our online method outperforms non-adaptive baselines,

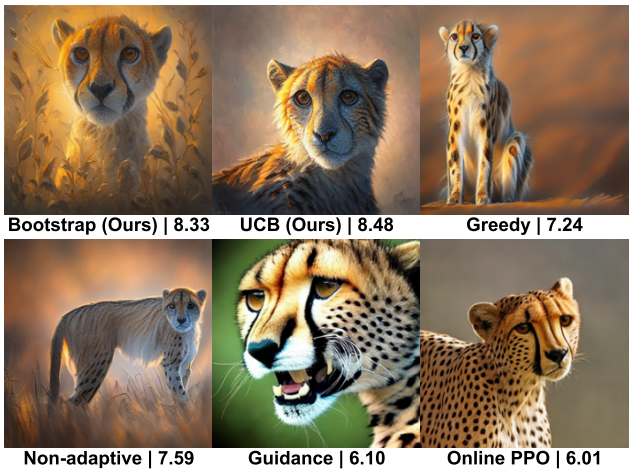

Figure 2. Images generated with aesthetic scores using the prompt "cheetah." Our methods outperform the baselines in terms of higher aesthetic scores while using the same amount of feedback.

underscoring the importance of adaptive data collection like our approach. Second, in comparison to online baselines such as **Online PPO** and **Greedy**, our approach consistently demonstrates superior performance. This highlights the effectiveness of our strategies, including (a) interleaving reward learning and diffusion model updates, (b) incorporating KL regularization and optimism for feedback-efficient fine-tuning. Third, among our methods, **Bootstrap** appears to be the most effective.

Remark 3 (Molecules)**.** *We do an analogous experiment in Section 7.1 to generate molecules with improved properties, specifically the Quantitative Estimate of Druglikeness (QED) score. For details, refer to Section D.2.*

7.2. Images

We validate the effectiveness of **SEIKO** in the image domain by fine-tuning for aesthetic quality. We employ Stable Diffusion v1.5 as our pre-trained model (Rombach et al., 2022). As for $r(x)$, we use the LAION Aesthetics Predictor V2 (Schuhmann, 2022), which is implemented as a linear MLP on the top of the OpenAI CLIP embeddings (Radford et al., 2021). This predictor is trained on a dataset comprising over 400k aesthetic ratings from 1 to 10, and has been used in existing works (Black et al., 2023). In our experiment, we set $M_1 = 1024$, $M_2 = 2048$, $M_3 = 4096$, and $M_4 = 8192$.

Results. We present the performances of **SEIKO** and baselines in Table 3. Some examples of generated images are shown in Figure 2. It is evident that **SEIKO** outperforms **Non-adaptive** and **Guidance** with better rewards, showcasing the necessity of adaptively collecting samples. Moreover, **SEIKO** achieves better performances even compared to **Greedy** and **Online PPO**. This again demonstrates that incorporating KL regularization and optimism are useful for feedback-efficient fine-tuning.

Table 3. Results for fine-tuning Stable Diffusion to optimize aesthetic scores. **SEIKO** attains high rewards within a fixed budget.

	Value ↑
Non-adaptive	7.22 ± 0.18
Guidance	5.78 ± 0.28
Online PPO	6.13 ± 0.33
Greedy	6.24 ± 0.28
UCB (Ours)	$\mathbf{8.17 \pm 0.33}$
Bootstrap (Ours)	7.77 ± 0.12

Finally, we present more qualitative results in Figure 5 to demonstrate the superiority of **SEIKO** over all baselines. For more additional images, refer to Appendix D.

8. Conclusion

This study introduces a novel feedback-efficient method to fine-tune diffusion models. Our approach is able to explore a novel region with a high reward on the manifold, which represents the feasible space. Our theoretical analysis highlights the effectiveness of our approach in terms of feedback efficiency, offering a specific regret guarantee.

As our future work, expanding on Section 7.1, we will explore the fine-tuning of recent diffusion models that are more customized for biological or chemical applications (Li et al., 2023; Gruver et al., 2023; Luo et al., 2022).

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Agarwal, A., N. Jiang, S. M. Kakade, and W. Sun (2019). Reinforcement learning: Theory and algorithms. *CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep*, 10–4.

Alamdari, S., N. Thakkar, R. van den Berg, A. X. Lu, N. Fusi, A. P. Amini, and K. K. Yang (2023). Protein generation with evolutionary diffusion: sequence is all you need. *bioRxiv*, 2023–09.

Albergo, M. S., N. M. Boffi, and E. Vanden-Eijnden (2023). Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*.

Avdeyev, P., C. Shi, Y. Tan, K. Dudnyk, and J. Zhou (2023,

23–29 Jul). Dirichlet diffusion score model for biological sequence generation. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett (Eds.), *Proceedings of the 40th International Conference on Machine Learning*, Volume 202 of *Proceedings of Machine Learning Research*, pp. 1276–1301. PMLR.

Bajusz, D., A. Rácz, and K. Héberger (2015). Why is tanimoto index an appropriate choice for fingerprint-based similarity calculations? *Journal of cheminformatics 7*(1), 1–13.

Balandat, M., B. Karrer, D. Jiang, S. Daulton, B. Letham, A. G. Wilson, and E. Bakshy (2020). BoTorch: A framework for efficient monte-carlo bayesian optimization. *Advances in neural information processing systems 33*, 21524–21538.

Bansal, A., H.-M. Chu, A. Schwarzschild, S. Sengupta, M. Goldblum, J. Geiping, and T. Goldstein (2023). Universal guidance for diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 843–852.

Black, K., M. Janner, Y. Du, I. Kostrikov, and S. Levine (2023). Training diffusion models with reinforcement learning. *arXiv preprint arXiv:2305.13301*.

Brantley, K., S. Dan, I. Gurevych, J.-U. Lee, F. Radlinski, H. Schütze, E. Simpson, and L. Yu (2021). Proceedings of the first workshop on interactive learning for natural language processing. In *Proceedings of the First Workshop on Interactive Learning for Natural Language Processing*.

Chang, J. D., M. Uehara, D. Sreenivas, R. Kidambi, and W. Sun (2021). Mitigating covariate shift in imitation learning via offline data without great coverage. *arXiv preprint arXiv:2106.03207*.

Chen, T., B. Xu, C. Zhang, and C. Guestrin (2016). Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*.

Chua, K., R. Calandra, R. McAllister, and S. Levine (2018). Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems 31*.

Clark, K., P. Vicol, K. Swersky, and D. J. Fleet (2023). Directly fine-tuning diffusion models on differentiable rewards. *arXiv preprint arXiv:2309.17400*.

Dhariwal, P. and A. Nichol (2021). Diffusion models beat gans on image synthesis. *Advances in neural information processing systems 34*, 8780–8794.

Dong, H., W. Xiong, D. Goyal, R. Pan, S. Diao, J. Zhang, K. Shum, and T. Zhang (2023). Raft: Reward ranked finetuning for generative foundation model alignment. *arXiv preprint arXiv:2304.06767*.

Efron, B. (1992). Bootstrap methods: another look at the jackknife. In *Breakthroughs in statistics: Methodology and distribution*, pp. 569–593. Springer.

Fan, Y., O. Watkins, Y. Du, H. Liu, M. Ryu, C. Boutilier, P. Abbeel, M. Ghavamzadeh, K. Lee, and K. Lee (2023). DPOK: Reinforcement learning for fine-tuning text-to-image diffusion models. *arXiv preprint arXiv:2305.16381*.

Frazier, P. I. (2018). A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*.

Gao, Z., Y. Han, Z. Ren, and Z. Zhou (2019). Batched multi-armed bandits problem. *Advances in Neural Information Processing Systems 32*.

Garnelo, M., J. Schwarz, D. Rosenbaum, F. Viola, D. J. Rezende, S. Eslami, and Y. W. Teh (2018). Neural processes. *arXiv preprint arXiv:1807.01622*.

Geist, M., B. Scherrer, and O. Pietquin (2019). A theory of regularized markov decision processes. In *International Conference on Machine Learning*, pp. 2160–2169. PMLR.

Ghari, P. M., A. Tseng, G. Eraslan, R. Lopez, T. Biancalani, G. Scalia, and E. Hajiramezanali (2023). Generative flow networks assisted biological sequence editing. In *NeurIPS 2023 Generative AI and Biology (GenBio) Workshop*.

Gómez-Bombarelli, R., J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, and A. Aspuru-Guzik (2018). Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science 4*(2), 268–276.

Graikos, A., N. Malkin, N. Jojic, and D. Samaras (2022). Diffusion models as plug-and-play priors. *Advances in Neural Information Processing Systems 35*, 14715–14728.

Gruslys, A., R. Munos, I. Danihelka, M. Lanctot, and A. Graves (2016). Memory-efficient backpropagation through time. *Advances in neural information processing systems 29*.

Gruver, N., S. Stanton, N. C. Frey, T. G. Rudner, I. Hotzel, J. Lafrance-Vanasse, A. Rajpal, K. Cho, and A. G. Wilson (2023). Protein design with guided discrete diffusion. *arXiv preprint arXiv:2305.20009*.

Ho, J., A. Jain, and P. Abbeel (2020). Denoising diffusion probabilistic models. *Advances in neural information processing systems 33*, 6840–6851.

Hu, E. J., Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen (2021). Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Igashov, I., H. Stärk, C. Vignac, V. G. Satorras, P. Frossard, M. Welling, M. Bronstein, and B. Correia (2022). Equivariant 3d-conditional diffusion models for molecular linker design. *arXiv preprint arXiv:2210.05274*.

Irwin, J. J. and B. K. Shoichet (2005). ZINC- a free database of commercially available compounds for virtual screening. *Journal of chemical information and modeling 45*(1), 177–182.

Jing, B., G. Corso, J. Chang, R. Barzilay, and T. Jaakkola (2022). Torsional diffusion for molecular conformer generation. *Advances in Neural Information Processing Systems 35*, 24240–24253.

Jo, J., S. Lee, and S. J. Hwang (2022). Score-based generative modeling of graphs via the system of stochastic differential equations. In *International Conference on Machine Learning*, pp. 10362–10383. PMLR.

Kingma, D. P. and J. Ba (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kingma, D. P. and M. Welling (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Krishnamoorthy, S., S. M. Mashkaria, and A. Grover (2023). Diffusion models for black-box optimization. *arXiv preprint arXiv:2306.07180*.

Krishnamurthy, A., J. Langford, A. Slivkins, and C. Zhang (2020). Contextual bandits with continuous actions: Smoothing, zooming, and adapting. *The Journal of Machine Learning Research 21*(1), 5402–5446.

Kveton, B., C. Szepesvari, S. Vaswani, Z. Wen, T. Lattimore, and M. Ghavamzadeh (2019). Garbage in, reward out: Bootstrapping exploration in multi-armed bandits. In *International Conference on Machine Learning*, pp. 3601–3610. PMLR.

Lattimore, T. and C. Szepesvári (2020). *Bandit algorithms*. Cambridge University Press.

Lee, K., H. Liu, M. Ryu, O. Watkins, Y. Du, C. Boutilier, P. Abbeel, M. Ghavamzadeh, and S. S. Gu (2023). Aligning text-to-image models using human feedback. *arXiv preprint arXiv:2302.12192*.

Li, Z., Y. Ni, T. A. B. Huygelen, A. Das, G. Xia, G.-B. Stan, and Y. Zhao (2023). Latent diffusion model for dna sequence generation. *arXiv preprint arXiv:2310.06150*.

Lipman, Y., R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le (2023). Flow matching for generative modeling. *ICLR 2023*.

Liu, G.-H., A. Vahdat, D.-A. Huang, E. A. Theodorou, W. Nie, and A. Anandkumar (2023). I2 sb: Image-to-image schr\" odinger bridge. *arXiv preprint arXiv:2302.05872*.

Liu, X., L. Wu, M. Ye, and Q. Liu (2022). Let us build bridges: Understanding and extending diffusion generative models. *arXiv preprint arXiv:2208.14699*.

Luo, S., Y. Su, X. Peng, S. Wang, J. Peng, and J. Ma (2022). Antigen-specific antibody design and optimization with diffusion-based generative models for protein structures. *Advances in Neural Information Processing Systems 35*, 9754–9767.

Neu, G., A. Jonsson, and V. Gómez (2017). A unified view of entropy-regularized markov decision processes. *arXiv preprint arXiv:1705.07798*.

Notin, P., J. M. Hernández-Lobato, and Y. Gal (2021). Improving black-box optimization in VAE latent space using decoder uncertainty. *Advances in Neural Information Processing Systems 34*, 802–814.

Osband, I., C. Blundell, A. Pritzel, and B. Van Roy (2016). Deep exploration via bootstrapped DQN. *Advances in neural information processing systems 29*.

Ouyang, L., J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. (2022). Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems 35*, 27730–27744.

Prabhudesai, M., A. Goyal, D. Pathak, and K. Fragkiadaki (2023). Aligning text-to-image diffusion models with reward backpropagation. *arXiv preprint arXiv:2310.03739*.

Qiu, S., L. Wang, C. Bai, Z. Yang, and Z. Wang (2022). Contrastive ucb: Provably efficient contrastive self-supervised learning in online reinforcement learning. In *International Conference on Machine Learning*, pp. 18168–18210. PMLR.

Radford, A., J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever (2021). Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*.

Rafailov, R., A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn (2024). Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems 36*.

Rombach, R., A. Blattmann, D. Lorenz, P. Esser, and B. Ommer (2022, June). High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695.

Sarkisyan, K. S., D. A. Bolotin, M. V. Meer, D. R. Usmanova, A. S. Mishin, G. V. Sharonov, D. N. Ivankov, N. G. Bozhanova, M. S. Baranov, O. Soylemez, et al. (2016). Local fitness landscape of the green fluorescent protein. *Nature 533*(7603), 397–401.

Schuhmann, C. (2022, Aug). LAION aesthetics.

Schulman, J., F. Wolski, P. Dhariwal, A. Radford, and O. Klimov (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Shi, Y., V. De Bortoli, A. Campbell, and A. Doucet (2023). Diffusion schr\" odinger bridge matching. *arXiv preprint arXiv:2303.16852*.

Somnath, V. R., M. Pariset, Y.-P. Hsieh, M. R. Martinez, A. Krause, and C. Bunne (2023). Aligned diffusion schr\" odinger bridges. *arXiv preprint arXiv:2302.11419*.

Song, J., C. Meng, and S. Ermon (2020). Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*.

Srinivas, N., A. Krause, S. M. Kakade, and M. Seeger (2009). Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*.

Tong, A., N. Malkin, G. Huguet, Y. Zhang, J. Rector-Brooks, K. Fatras, G. Wolf, and Y. Bengio (2023). Improving and generalizing flow-based generative models with minibatch optimal transport. *arXiv preprint arXiv:2302.00482*.

Touvron, H., L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al. (2023). Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Trabucco, B., X. Geng, A. Kumar, and S. Levine (2022). Design-bench: Benchmarks for data-driven offline model-based optimization. In *International Conference on Machine Learning*, pp. 21658–21676. PMLR.

Tseng, A. M., N. L. Diamant, T. Biancalani, and G. Scalia (2023). GraphGUIDE: interpretable and controllable conditional graph generation with discrete bernoulli diffusion. In *ICLR 2023 - Machine Learning for Drug Discovery workshop*.

Uehara, M., Y. Zhao, K. Black, E. Hajiramezanali, G. Scalia, N. L. Diamant, A. M. Tseng, T. Biancalani, and S. Levine (2024). Fine-tuning of continuous-time diffusion models as entropy-regularized control. *arXiv preprint arXiv:2402.15194*.

Valko, M., N. Korda, R. Munos, I. Flaounas, and N. Cristianini (2013). Finite-time analysis of kernelised contextual bandits. *arXiv preprint arXiv:1309.6869*.

Vignac, C., I. Krawczuk, A. Siraudin, B. Wang, V. Cevher, and P. Frossard (2023). Digress: Discrete denoising diffusion for graph generation. In *The Eleventh International Conference on Learning Representations*.

Wallace, B., M. Dang, R. Rafailov, L. Zhou, A. Lou, S. Purushwalkam, S. Ermon, C. Xiong, S. Joty, and N. Naik (2023). Diffusion model alignment using direct preference optimization. *arXiv preprint arXiv:2311.12908*.

Wang, Z., J. J. Hunt, and M. Zhou (2022). Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*.

Watson, J. L., D. Juergens, N. R. Bennett, B. L. Trippe, J. Yim, H. E. Eisenach, W. Ahern, A. J. Borst, R. J. Ragotte, L. F. Milles, et al. (2023). De novo design of protein structure and function with rfdiffusion. *Nature 620*(7976), 1089–1100.

Wu, X., K. Sun, F. Zhu, R. Zhao, and H. Li (2023). Better aligning text-to-image models with human preference. *arXiv preprint arXiv:2303.14420*.

Xu, J., X. Liu, Y. Wu, Y. Tong, Q. Li, M. Ding, J. Tang, and Y. Dong (2023). Imagereward: Learning and evaluating human preferences for text-to-image generation. *arXiv preprint arXiv:2304.05977*.

Yang, K., J. Tao, J. Lyu, C. Ge, J. Chen, Q. Li, W. Shen, X. Zhu, and X. Li (2023). Using human feedback to fine-tune diffusion models without any reward model. *arXiv preprint arXiv:2311.13231*.

Yuan, H., K. Huang, C. Ni, M. Chen, and M. Wang (2023). Reward-directed conditional diffusion: Provable distribution estimation and reward improvement. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Zhang, T., T. Ren, M. Yang, J. Gonzalez, D. Schuurmans, and B. Dai (2022). Making linear MDPs practical via contrastive representation learning. In *International Conference on Machine Learning*, pp. 26447–26466. PMLR.

Zhang, Z., E. Strubell, and E. Hovy (2022). A survey of active learning for natural language processing. *arXiv preprint arXiv:2210.10109*.

Zhang, Z., M. Xu, A. C. Lozano, V. Chenthamarakshan, P. Das, and J. Tang (2024). Pre-training protein encoder via siamese sequence-structure diffusion trajectory prediction. *Advances in Neural Information Processing Systems 36*.

Zhou, D., L. Li, and Q. Gu (2020). Neural contextual bandits with ucb-based exploration. In *International Conference on Machine Learning*, pp. 11492–11502. PMLR.

A. Implementation Details of Planning Algorithms

We seek to solve the following optimization problem:

$$f^{(i)}, \nu^{(i)} = \underset{f:[0,T]\times\mathcal{X}\to\mathcal{X}, \nu\in\Delta(\mathcal{X})}{\operatorname{argmax}} \underbrace{\mathbb{E}_{\mathbb{P}^{f,\nu}}[(\hat{r}^{(i)} + \hat{g}^{(i)})(x_T)]}_{\text{(B) Optimistic reward}}$$

$$- \alpha \underbrace{\mathbb{E}_{\mathbb{P}^{f,\nu}}\left[\log\left(\frac{\nu(x_0)}{\nu^{\text{pre}}(x_0)}\right) + \int_{t=0}^{T} \frac{\|g^{(0)}(t,x_t)\|^2}{2\sigma^2(t)} dt\right]}_{\text{(A1) KL Regularization relative to a diffusion at iteration 0}} - \beta_{i-1} \underbrace{\mathbb{E}_{\mathbb{P}^{f,\nu}}\left[\log\left(\frac{\nu(x_0)}{\nu^{(i-1)}(x_0)}\right) + \int_{t=0}^{T} \frac{\|g^{(i-1)}(t,x_t)\|^2}{2\sigma^2(t)} dt\right]}_{\text{(A2) KL Regularization relative to a diffusion at iteration } i-1}$$

For simplicity, we fix $\nu^{(i)} = \nu_{\text{pre}}$. For methods to optimize initial distributions, refer to Uehara et al. (2024).

Here, suppose that f is parameterized by θ (e.g., neural networks). Then, we update θ using stochastic gradient descent. Consider iteration $i \in [1, 2, \cdots,]$. With parameter θ fixed in $f(t, x_t; \theta)$, by simulating an SDE with

$$dx_t = f(t, x_t; \theta)dt + \sigma(t)dw_t, x_0 \sim \nu^{\text{pre}},$$
$$dz_t = \frac{\|g^{(0)}(t, x_t; \theta)\|^2}{2\sigma^2(t)} dt, \quad dZ_t = \frac{\|g^{(i-1)}(t, x_t; \theta)\|^2}{2\sigma^2(t)} dt,$$

we obtain n trajectories

$$\{x_0^{\langle k\rangle}, \cdots, x_T^{\langle k\rangle}\}_{k=1}^n, \{z_0^{\langle k\rangle}, \cdots, z_T^{\langle k\rangle}\}_{k=1}^n, \{Z_0^{\langle k\rangle}, \cdots, Z_T^{\langle k\rangle}\}_{k=1}^n.$$

It is possible to use any off-the-shelf discretization methods, such as the Euler–Maruyama method. For instance, starting from $x_0^{\langle k\rangle} \sim \nu$, a trajectory can be obtained as follows:

$$x_t^{\langle k\rangle} = x_{t-1}^{\langle k\rangle} + f(t-1, x_{t-1}^{\langle k\rangle}; \theta)\Delta t + \sigma(t)(\Delta w_t), \quad \Delta w_t \sim \mathcal{N}(0, (\Delta t)^2),$$
$$z_t^{\langle k\rangle} = z_{t-1}^{\langle k\rangle} + \frac{\|f(t-1, x_{t-1}^{\langle k\rangle}; \theta) - f^{\text{pre}}(t-1, x_{t-1}^{\langle k\rangle})\|^2}{2\sigma^2(t-1)} \Delta t,$$
$$Z_t^{\langle k\rangle} = Z_{t-1}^{\langle k\rangle} + \frac{\|f(t-1, x_{t-1}^{\langle k\rangle}; \theta) - f^{(i-1)}(t-1, x_{t-1}^{\langle k\rangle}; \theta^{(i-1)})\|^2}{2\sigma^2(t-1)} \Delta t.$$

Finally, using automatic differentiation, we update θ as follows:

$$\theta_{i+1} = \theta_i - \rho \nabla_\theta \left\{ \frac{1}{n} \sum_{k=1}^{n} L(x_T^{\langle k\rangle}) - \alpha z_T^{\langle k\rangle} - \beta_{i-1} Z_T^{\langle k\rangle} \right\} \Bigg|_{\theta=\theta_i},$$

where ρ is a learning rate. For the practical selection of the learning rate ρ, we use the Adam optimizer (Kingma and Ba, 2014) in this step.

Note when discretization steps are large, the above calculation might be too computationally intensive. Regarding tricks to mitigate it, refer to (Clark et al., 2023; Prabhudesai et al., 2023).

B. Proofs

B.1. Preliminary

In this section, we want to show how Assumption 1 holds when using linear models. A similar argument still holds when using GPs (Valko et al., 2013; Srinivas et al., 2009).

Suppose we have a linear model:

$$\forall x \in \mathcal{X}_{\text{pre}};\ y = \mu^\top \phi(x) + \epsilon$$

where $\|\mu\|_2, \|\phi(x)\|_2 \le B$, and ϵ is a σ-sub-Gaussian noise. Then, the ridge estimator is defined as

$$\hat{\mu} := \Sigma_i^{-1} \sum_{j=1}^{i} \phi(x^{(j)}) y^{(j)}, \quad \Sigma_i := \sum_{j=1}^{i} \phi(x^{(j)}) \phi^\top(x^{(j)}) + \lambda I.$$

With probability $1-\delta$, we can show that

$$\forall x \in \mathcal{X}_{\text{pre}};\ \langle \phi(x), \hat{\mu} - \mu \rangle \leq C_1(\delta) \|\phi(x)\|_{\Sigma_i}$$

where

$$C_1(\delta) := \left\{ B\sqrt{\lambda} + \sqrt{\sigma^2 \left(\log(1/\delta^2) + d \log\left(1 + \frac{KB^2}{d\lambda}\right)\right)} \right\}.$$

This is proved as follows. First, by some algebra, we have

$$\hat{\mu} - \mu = \lambda \Sigma_i^{-1} \mu + \Sigma_i^{-1} \sum_{j=1}^{i} \phi(x^{(j)}) \epsilon_j.$$

Hence, with probability $1-\delta$, we can show that

$$\begin{aligned}
\hat{r}(x) - r(x) &= \langle \phi(x), \hat{\mu} - \mu \rangle \\
&= \left\langle \phi(x), \lambda \Sigma_i^{-1} \mu + \Sigma_i^{-1} \sum_j \phi(x^{(j)}) \epsilon_j \right\rangle \\
&\leq \|\phi(x)\|_{\Sigma_i} \|\lambda \mu\|_{\Sigma_i^{-1}} + \|\phi(x)\|_{\Sigma_i} \| \sum_j \phi(x^{(j)}) \epsilon_j \|_{\Sigma_i^{-1}} && \text{(CS inequality)} \\
&\leq \|\phi(x)\|_{\Sigma_i} \left\{ B\sqrt{\lambda} + \sqrt{\sigma^2 \left(\log(1/\delta^2) + d \log\left(1 + \frac{iB^2}{d\lambda}\right)\right)} \right\}. \\
& && \text{(Use Proof of Prop 6.7 and Lemma A.9 in Agarwal et al. (2019))}
\end{aligned}$$

Hence, we can set

$$C_1(\delta) = B\sqrt{\lambda} + \sqrt{\sigma^2 \left(\log(1/\delta^2) + d \log\left(1 + \frac{KB^2}{d\lambda}\right)\right)}.$$

B.2. Proof of Theorem 1

To simplify the notation, we let $f^{(i)}(x) = \hat{r}^{(i)}(x) + \hat{g}^{(i)}(x)$.

We first note that

$$\text{Term(A1)} = \text{KL}(\mathbb{P}^{f,\nu}(\cdot) | \mathbb{P}^{f^{\text{pre}},\nu^{\text{pre}}}(\cdot)).$$

Letting $\mathbb{P}^{f,\nu}_{\cdot|0}$ be the distribution conditioned on an initial state x_0 (hence, ν does not matter), this is because

$$\begin{aligned}
\text{KL}(\mathbb{P}^{f,\nu}(\cdot) \| \mathbb{P}^{f^{\text{pre}},\nu^{\text{pre}}}(\cdot)) &= \text{KL}(\nu \| \nu^{\text{pre}}) + \mathbb{E}_{x_0 \sim \nu}[\text{KL}(\mathbb{P}^{f,\nu}_{\cdot|0}(\cdot|x_0) \| \mathbb{P}^{f^{\text{pre}},\nu^{\text{pre}}}_{\cdot|0}(\cdot|x_0))] \\
&= \text{KL}(\nu \| \nu^{\text{pre}}) + \mathbb{E}_{x_0 \sim \nu} \left[\mathbb{E}_{\mathbb{P}^{f,\nu}_{\cdot|0}(\cdot|x_0)} \left[\log \left(\frac{d\mathbb{P}^{f,\nu}_{\cdot|0}(\cdot|x_0)}{d\mathbb{P}^{f^{\text{pre}},\nu^{\text{pre}}}_{\cdot|0}(\cdot|x_0))} \right) \right] \right] \\
&= \text{KL}(\nu \| \nu^{\text{pre}}) + \mathbb{E}_{\mathbb{P}^{f,\nu}} \left[\int_{t=0}^{T} \frac{\|(f - f^{\text{pre}})(t, x_t)\|_2^2}{2\sigma^2(t)} dt \right].
\end{aligned}$$

Similarly, we have

$$\text{Term(A2)} = \text{KL}(\mathbb{P}^{f,\nu}(\cdot) | \mathbb{P}^{f^{(i)},\nu^{(i)}}(\cdot)).$$

Therefore, the objective function in (4) becomes

$$\mathbb{E}_{\mathbb{P}^{f,\nu}}[r(x_T)] - \alpha \text{KL}(\mathbb{P}^{f,\nu}(\cdot) \| \mathbb{P}^{f^{\text{pre}},\nu^{\text{pre}}}(\cdot)) - \beta \text{KL}(\mathbb{P}^{f,\nu}(\cdot) \| \mathbb{P}^{f^{(i-1)},\nu^{(i-1)}}(\cdot)).$$

Now, we further aim to modify the above objective function in (4). Here, letting $\mathbb{P}^{f,\nu}_{\cdot|T}(\cdot|x_T)$ be the conditional distribution over $\mathcal{C}$ conditioning on a state x_T, we have

$$\begin{aligned}\mathrm{KL}(\mathbb{P}^{f,\nu}(\cdot)\|\mathbb{P}^{f^{\mathrm{pre}},\nu^{\mathrm{pre}}}(\cdot)) &= \int \log\left(\frac{d\mathbb{P}^{f,\nu}(\cdot)}{d\mathbb{P}^{f^{\mathrm{pre}},\nu^{\mathrm{pre}}}(\cdot)}\right) d\mathbb{P}^{f,\nu}(\cdot)\\ &= \int \log\left\{\left(\frac{d\mathbb{P}^{f,\nu}_T(x_T)}{d\mathbb{P}^{f^{\mathrm{pre}},\nu^{\mathrm{pre}}}_T(x_T)}\right) + \left(\frac{d\mathbb{P}^{f,\nu}_{\cdot|T}(\tau|x_T)}{d\mathbb{P}^{f^{\mathrm{pre}},\nu^{\mathrm{pre}}}_{\cdot|T}(\tau|x_T)}\right)\right\} d\mathbb{P}^{f,\nu}(\tau)\\ &= \mathrm{KL}(\mathbb{P}^{f,\nu}_T\|\mathbb{P}^{f^{\mathrm{pre}},\nu^{\mathrm{pre}}}_T) + \mathbb{E}_{x_T\sim\mathbb{P}^{f,\nu}_T}[\mathrm{KL}(\mathbb{P}^{f,\nu}_{\cdot|T}(\tau|x_T)\|\mathbb{P}^{f^{\mathrm{pre}},\nu^{\mathrm{pre}}}_{\cdot|T}(\tau|x_T))].\end{aligned}$$

Hence, the objective function in (4) becomes

$$\begin{aligned}&\underbrace{\mathbb{E}_{x_T\sim\mathbb{P}^{f,\nu}_T}[r(x_T)] - \alpha\mathrm{KL}(\mathbb{P}^{f,\nu}_T\|\mathbb{P}^{f^{\mathrm{pre}},\nu^{\mathrm{pre}}}_T) - \beta\mathrm{KL}(\mathbb{P}^{f,\nu}_T\|\mathbb{P}^{f^{(i-1)},\nu^{(i-1)}}_T)}_{\text{(i)}}\\ &\underbrace{-\alpha\mathbb{E}_{x_T\sim\mathbb{P}^{f,\nu}_T}[\mathrm{KL}(\mathbb{P}^{f,\nu}_{\cdot|T}(\cdot|x_T)\|\mathbb{P}^{f^{\mathrm{pre}},\nu^{\mathrm{pre}}}_{\cdot|T}(\cdot|x_T))]}_{\text{(ii)}}\underbrace{-\beta\mathbb{E}_{x_T\sim\mathbb{P}^{f,\nu}_T}[\mathrm{KL}(\mathbb{P}^{f,\nu}_{\cdot|T}(\cdot|x_T)\|\mathbb{P}^{f^{(i-1)},\nu^{(i-1)}}_{\cdot|T}(\cdot|x_T))]}_{\text{(iii)}}.\end{aligned} \tag{9}$$

From now on, we use the induction method. Suppose we have

$$\mathbb{P}^{f^{(i-1)},\nu^{(i-1)}}_{\cdot|T}(\cdot|x_T) = \mathbb{P}^{f^{\mathrm{pre}},\nu^{\mathrm{pre}}}_{\cdot|T}(\cdot|x_T). \tag{10}$$

Indeed, this holds when $i = 1$. Assuming the above holds at $i-1$, the objective function in (4) becomes

$$\begin{aligned}&\underbrace{\mathbb{E}_{x_T\sim\mathbb{P}^{f,\nu}_T}[r(x_T)] - \alpha\mathrm{KL}(\mathbb{P}^{f,\nu}_T\|\mathbb{P}^{f^{\mathrm{pre}},\nu^{\mathrm{pre}}}_T) - \beta\mathrm{KL}(\mathbb{P}^{f,\nu}_T\|\mathbb{P}^{f^{(i-1)},\nu^{(i-1)}}_T)}_{\text{(i)}}\\ &\underbrace{-\alpha\mathbb{E}_{x_T\sim\mathbb{P}^{f,\nu}_T}[\mathrm{KL}(\mathbb{P}^{f,\nu}_{\cdot|T}(\cdot|x_T)\|\mathbb{P}^{f^{\mathrm{pre}},\nu^{\mathrm{pre}}}_{\cdot|T}(\cdot|x_T))]}_{\text{(ii)}}\underbrace{-\beta\mathbb{E}_{x_T\sim\mathbb{P}^{f,\nu}_T}[\mathrm{KL}(\mathbb{P}^{f,\nu}_{\cdot|T}(\cdot|x_T)\|\mathbb{P}^{f^{\mathrm{pre}},\nu^{\mathrm{pre}}}_{\cdot|T}(\cdot|x_T))]}_{\text{(iii)}}.\end{aligned} \tag{11}$$

By maximizing each term (i), (ii), (iii) in (11) over $\mathbb{P}^{f,\nu}(\cdot,\diamond) = \mathbb{P}^{f,\nu}_T(\diamond)\times\mathbb{P}^{f,\nu}_{\cdot|T}(\cdot|\diamond)$, we get

$$\begin{aligned}\mathbb{P}^{f^{(i)},\nu^{(i)}}_T(\diamond) &\propto \exp\left(\frac{\hat{r}^{(i)}(\diamond)+\hat{g}^{(i)}(\diamond)}{\alpha+\beta}\right)\{p^{(i-1)}(\diamond)\}^{\frac{\beta}{\alpha+\beta}}\{p_{\mathrm{pre}}(\diamond)\}^{\frac{\alpha}{\alpha+\beta}},\\ \mathbb{P}^{f^{(i)},\nu^{(i)}}_{\cdot|T}(\cdot|x_T) &= \mathbb{P}^{f^{\mathrm{pre}},\nu^{\mathrm{pre}}}_{\cdot|T}(\cdot|x_T).\end{aligned} \tag{12}$$

Hence, the induction in (10) holds, and the statement is concluded.

Remark 4. *Note from the above, we can conclude that the whole distribution on $\mathcal{C}$ is*

$$\frac{1}{C}\exp\left(\frac{\hat{r}^{(i)}(x_T)+\hat{g}^{(i)}(x_T)}{\alpha+\beta}\right)\{p^{(i-1)}(\tau)\}^{\frac{\beta}{\alpha+\beta}}\{p_{\mathrm{pre}}(\tau)\}^{\frac{\alpha}{\alpha+\beta}}.$$

Remark 5. *Some readers might wonder in the part we optimize over $\mathbb{P}_{f,\nu}$ rather than f,ν in the final step. Indeed, this step would go through when we use non-Markovian drifts for f. While we use Markovian drifts, this is still fine because the optimal drift must be Markovian even when we optimize over non-Markovian drifts. We have chosen to present this intuitive proof first in order to more clearly convey our message regarding the bridge-preserving property (12). We will formalize it in Theorem 4 in Section C.*

B.3. Proof of Theorem 2

In the following, we condition on an event where

$$\forall i\in[K];\ x\in\mathcal{X}_{\mathrm{pre}}\ : |\hat{r}^{(i)}(x) - r(x)| \le \hat{g}^{(i)}(x).$$

Then, we define

$$\hat{f}^{(i)}(x) := \hat{r}^{(i)}(x) + \hat{g}^{(i)}(x).$$

Recall that we have

$$\forall x \in \mathcal{X}; r(x) \leq \hat{r}^{(i)}(x) + \hat{g}^{(i)}(x) = \hat{f}^{(i)}(x). \tag{13}$$

because when $x \in \mathcal{X}_\rho$, it is from the assumption and when $x \notin \mathcal{X}_\rho$, $r(x)$ takes 0. Furthermore, we have

$$\forall x \in \mathcal{X}_{\text{pre}};\ \hat{f}^{(i)}(x) - r(x) \leq \hat{g}^{(i)}(x) + \hat{r}^{(i)}(x) - r(x) \leq 2\hat{g}^{(i)}(x). \tag{14}$$

With the above preparation, from now on, we aim to show the regret guarantee. First, by some algebra, we have

$$\begin{aligned}
&1/K \sum_{i=1}^{K} \mathbb{E}_{x\sim\pi}[r(x)] - \mathbb{E}_{x\sim p^{(i)}}[r(x)] \\
&= 1/K \sum_{i=1}^{K} \mathbb{E}_{x\sim\pi}[r(x)] - \mathbb{E}_{x\sim\pi}[\hat{f}^{(i)}(x)] + \mathbb{E}_{x\sim\pi}[\hat{f}^{(i)}(x)] - \mathbb{E}_{x\sim p^{(i)}}[\hat{f}^{(i)}(x)] + \mathbb{E}_{x\sim p^{(i)}}[\hat{f}^{(i)}(x)] - \mathbb{E}_{x\sim p^{(i)}}[r(x)] \\
&\leq 1/K \sum_{i=1}^{K} \mathbb{E}_{x\sim\pi}[\hat{f}^{(i)}(x)] - \mathbb{E}_{x\sim p^{(i)}}[\hat{f}^{(i)}(x)] + \mathbb{E}_{x\sim p^{(i)}}[\hat{f}^{(i)}(x)] - \mathbb{E}_{x\sim p^{(i)}}[r(x)] && \text{(Optimism (13))} \\
&\leq 1/K \sum_{i=1}^{K} \mathbb{E}_{x\sim\pi}[\hat{f}^{(i)}(x)] - \mathbb{E}_{x\sim p^{(i)}}[\hat{f}^{(i)}(x)] + 2\mathbb{E}_{x\sim p^{(i)}}[\hat{g}^{(i)}(x;\mathcal{D}_i)]. \\
& && \text{(Use (14) recalling the support of } p^{(i)} \text{ is in } \mathcal{X}_{\text{pre}})
\end{aligned}$$

Therefore, we have

$$\begin{aligned}
&J_\alpha(\pi) - \frac{1}{K}\sum_i J_\alpha(p^{(i)}) \\
&= \frac{1}{K}\sum_{i=1}^{K} \underbrace{\mathbb{E}_{x\sim\pi}[\hat{f}^{(i)}(x)] - \mathbb{E}_{x\sim p^{(i)}}[\hat{f}^{(i)}(x)] - \alpha\text{KL}(\pi\|\rho_{\text{pre}}) + \alpha\text{KL}(p^{(i)}\|\rho_{\text{pre}})}_{\text{(To)}} + 2\mathbb{E}_{x\sim p^{(i)}}[\hat{g}^{(i)}(x;\mathcal{D}_i)].
\end{aligned}$$

Now, we analyze the term (To):

$$\begin{aligned}
&\sum_{i=1}^{K} \mathbb{E}_{x\sim\pi}[\hat{f}^{(i)}(x)] - \mathbb{E}_{x\sim p^{(i)}}[\hat{f}^{(i)}(x)] - \alpha\text{KL}(\pi\|\rho_{\text{pre}}) + \alpha\text{KL}(p^{(i)}\|\rho_{\text{pre}}) \\
&= \sum_{i=1}^{K} \beta_{i-1}\mathbb{E}_{x\sim\pi}[\log(\pi(x)/p^{(i-1)}(x))] - (\beta_{i-1}+\alpha)\mathbb{E}_{x\sim\pi}[\log(\pi(x)/p^{(i)}(x))] - \beta_{i-1}\mathbb{E}_{x\sim p^{(i)}}[\log(p^{(i)}(x)/p^{(i-1)}(x))] \\
&\leq \sum_{i=1}^{K} \beta_{i-1}\mathbb{E}_{x\sim\pi}[\log(\pi(x)/p^{(i-1)}(x))] - (\beta_{i-1}+\alpha)\mathbb{E}_{x\sim\pi}[\log(\pi(x)/p^{(i)}(x))] \\
&\leq \beta_0\mathbb{E}_{x\sim\pi}[\log(\pi(x)/p^{(0)}(x))] \\
&= 0. && \text{(Recall we set } \beta_{i-1} = (i-1)\alpha)
\end{aligned}$$

Here, from the second line to the third line, we use

$$\begin{aligned}
&\mathbb{E}_{x\sim\pi}[\hat{f}^{(i)}(x)] - \alpha\text{KL}(\pi\|\rho_{\text{pre}}) - \beta_{i-1}\text{KL}(\pi\|p^{(i-1)}) - \mathbb{E}_{x\sim p^{(i)}}[\hat{f}^{(i)}(x)] + \alpha\text{KL}(p^{(i)}\|\rho_{\text{pre}}) + \beta_{i-1}\text{KL}(p^{(i)}\|p^{(i-1)}) \\
&= \mathbb{E}_{x\sim\pi}[\hat{f}^{(i)}(x)] - \alpha\text{KL}(\pi\|\rho_{\text{pre}}) - \beta_{i-1}\text{KL}(\pi\|p^{(i-1)}) - (\alpha+\beta_{i-1})\int \hat{f}^{(i)}(x)\{\rho_{\text{pre}}(x)\}^{\alpha/\gamma}\{p^{(i-1)}(x)\}^{\beta_{i-1}/\gamma}dx \\
&= (\beta_{i-1}+\alpha)\mathbb{E}_{x\sim\pi}[\log(\pi(x)/p^{(i)}(x))].
\end{aligned}$$

Combining results so far, the proof is concluded. Therefore, we have

$$J_\alpha(\pi) - \frac{1}{K}\sum_i J_\alpha(p^{(i)}) \leq \frac{2}{K}\sum_i \mathbb{E}_{x\sim p^{(i)}}[\hat{g}^{(i)}(x;\mathcal{D}_i)].$$

B.4. Proof of Lemma 1

Refer to the proof of Proposition 6.7 in Agarwal et al. (2019) in detail. For completeness, we sketch the proof here. We denote

$$\Sigma_i := \lambda I + \sum_i \phi(x^{(i)})\phi(x^{(i)})^\top,\ \hat{h}^{(i)}(x) := \min\left(1, \sqrt{\phi(\cdot)^\top(\Sigma_i + \lambda I)^{-1}\phi(\cdot)}\right)$$

Here, condoning on the event where Assumption 1 holds, we have

$$\begin{aligned}
\underbrace{\mathbb{E}\left[\frac{1}{K}\sum_i \mathbb{E}_{x\sim p^{(i)}}[\hat{g}^{(i)}(x)]\right]}_{\textit{Expected regret}} &\le 1/\sqrt{K} \times \sqrt{\sum_i \mathbb{E}[\mathbb{E}_{x\sim p^{(i)}}[\{\hat{g}^{(i)}(x)\}^2]]} && \text{(Jensen's inequality)}\\
&= \frac{C_1(\delta)}{\sqrt{K}} \times \sqrt{\sum_i \mathbb{E}[\mathbb{E}_{x\sim p^{(i)}}[\log(1+\{\hat{h}^{(i)}(x)\}^2)]]} && (\ x \le 2\log(1+x) \text{ when } 1 > x > 0)\\
&= \frac{C_1(\delta)}{\sqrt{K}} \times \sqrt{2\sum_i \mathbb{E}\left[\log\left(1+\phi^\top(x^{(i)})(\Sigma_i+\lambda)^{-1}\phi(x^{(i)})\right)\right]}\\
&= \frac{C_1(\delta)}{\sqrt{K}} \times \sqrt{2\sum_i \mathbb{E}[\log(\det\Sigma_i/\det\Sigma_{i-1})]} && \text{(Lemma 6.10 in Agarwal et al. (2019))}\\
&= \frac{C_1(\delta)}{\sqrt{K}} \times \sqrt{2\mathbb{E}[\log(\det\Sigma_K/\det\Sigma_0)]} && \text{(Telescoping sum)}\\
&\le \frac{C_1(\delta)}{\sqrt{K}} \times \sqrt{2d\log\left(1+\frac{KB^2}{d\lambda}\right)} && \text{(Elliptical potential lemma)}
\end{aligned}$$

By taking $\delta = 1/K^2$ and recalling

$$C_1(\delta) = B\sqrt{\lambda} + \sqrt{\sigma^2\left(\log(1/\delta^2) + d\log\left(1+\frac{KB^2}{d\lambda}\right)\right)},$$

the statement is concluded.

C. More Detailed Theoretical Properties

In this section, we show more detailed theoretical properties of fine-tuned diffusion models. In this proof, to simplify the notation, we denote β_{i-1} by β.

As a first step, we present a more analytical form of the optimal drift term by invoking the HJB equation, and the optimal value function. Note that we define the optimal value function by

$$v_t^\star(x_t) = \mathbb{E}_{\tau\sim\mathbb{P}^{(i)}}\left[\hat{r}^{(i)}(x_T) + \hat{g}^{(i)}(x_T) - \int_{s=t}^T \frac{\beta\|(f^{(i)} - f^{(i-1)})(s,x_s)\|_2^2 + \alpha\|(f^{\text{pre}} - f^{(i)})(s,x_s)\|_2^2}{2\sigma^2(s)}ds|x_t\right]$$

Theorem 3 (Optimal drift and value function)**.** *The optimal drift satisfies*

$$u^\star(t,x) = \sigma^2(t)\frac{\nabla_x v_t^\star(x)}{\alpha+\beta} + \frac{\{\alpha f^{\text{pre}} + \beta f^{(i-1)}\}(t,x)}{\alpha+\beta} \tag{15}$$

where

$$\exp\left(\frac{v_t^\star(x)}{\alpha+\beta}\right) = \mathbb{E}_{\mathbb{P}^{\bar{f}}}\left[\exp\left(\frac{(\hat{r}^{(i)}+\hat{g}^{(i)})(x_T)}{\alpha+\beta} - \alpha\beta\int_{s=t}^T \frac{\|(f^{(i-1)} - f^{\text{pre}})(s,x_s)\|^2}{2\sigma^2(s)}ds\right)|x_t = x\right]$$

and the measure $\mathbb{P}^{\bar{f}}$ is induced by the following SDE:

$$dx_t = \bar{f}(t,x_t)dt + \sigma(t)dw_t, \quad \bar{f} := \frac{\beta f^{(i-1)} + \alpha f^{\text{pre}}}{\alpha+\beta}, \quad \bar{\nu} := \frac{\beta\nu^{(i-1)} + \alpha\nu^{\text{pre}}}{\alpha+\beta}.$$

Using the above characterization, we can calculate the distribution induced by the optimal drift and initial distribution.

Theorem 4 (Formal statement of Theorem 1)**.** *Let $\mathbb{P}^{(i)}(\cdot)$ be a distribution over trajectories on $\mathcal{C}$ induced by the diffusion model governed by $f^{(i)}, \nu^{(i)}$. Similarly, we define the conditional distribution over $\mathcal{C}$ condoning on the terminal state x_T by $\mathbb{P}^{(i)}_{\cdot|T}(\cdot|x_T)$. Then, the following holds:*

$$\begin{aligned}
\mathbb{P}^{(i)}(\tau) &= \exp\left(\frac{\hat{r}^{(i)}(x_T) + \hat{g}^{(i)}(x_T)}{\alpha+\beta}\right)\{\mathbb{P}^{(i-1)}(\tau)\}^{\frac{\beta}{\alpha+\beta}}\{\mathbb{P}_{\mathrm{pre}}(\tau)\}^{\frac{\alpha}{\alpha+\beta}},\\
\mathbb{P}^{(i)}_{\cdot|T}(\tau|x_T) &= \mathbb{P}^{\mathrm{pre}}_{\cdot|T}(\tau|x_T),\\
p^{(i)}(x_T) &\propto \exp\left(\frac{\hat{r}^{(i)}(x_T) + \hat{g}^{(i)}(x_T)}{\alpha+\beta}\right)\{p^{(i-1)}(x_T)\}^{\frac{\beta}{\alpha+\beta}}\{p_{\mathrm{pre}}(x_T)\}^{\frac{\alpha}{\alpha+\beta}}.
\end{aligned}$$

C.1. Proof of Theorem 3

From the Hamilton–Jacobi–Bellman (HJB) equation, we have

$$\max_u \left\{\frac{\sigma^2(t)}{2}\sum_{i,j}\frac{d^2 v_t^\star(x)}{dx^{[i]}dx^{[j]}} + u\cdot\nabla v_t^\star(x) + \frac{dv_t^\star(x)}{dt} - \frac{\alpha\|(u - f^{\mathrm{pre}})(t,x_t)\|_2^2}{2\sigma^2(t)} - \frac{\beta\|(u - f^{(i-1)})(t,x_t)\|_2^2}{2\sigma^2(t)}\right\} = 0.$$

Here, we analyze the following terms:

$$(a) = u\cdot\left\{\nabla v_t^\star(x) + \frac{\alpha f^{\mathrm{pre}}(t,x_t) + \beta f^{(i-1)}(t,x_t)}{\sigma^2(t)}\right\} - \frac{\alpha+\beta}{2\sigma^2(t)}\|u\|_2^2 + \frac{dv_t^\star(x)}{dt} - \frac{\alpha\|f^{\mathrm{pre}}(t,x_t)\|_2^2 + \beta\|f^{(i-1)}(t,x_t)\|_2^2}{2\sigma^2(t)}.$$

This is equal to

$$\begin{aligned}
(a) &= -\frac{\alpha+\beta}{2\sigma^2(t)}\left\|u - \frac{\sigma^2(t)\nabla v_t^\star(x)}{\alpha+\beta} - \frac{\alpha f^{\mathrm{pre}} + \beta f^{(i-1)}}{\alpha+\beta}\right\|^2 + (b),\\
(b) &= \frac{dv_t^\star(x)}{dt} - \frac{\alpha\|f^{\mathrm{pre}}\|_2^2 + \beta\|f^{(i-1)}\|_2^2}{2\sigma^2(t)} + \frac{\alpha+\beta}{2\sigma^2(t)}\left\|\frac{\sigma^2(t)\nabla v_t^\star(x)}{\alpha+\beta} + \frac{\alpha f^{\mathrm{pre}} + \beta f^{(i-1)}}{\alpha+\beta}\right\|^2\\
&= \frac{dv_t^\star(x)}{dt} + \frac{\sigma^2(t)\|\nabla v_t^\star(x)\|_2^2}{2(\alpha+\beta)} - \frac{\alpha\beta}{2\sigma^2(t)(\alpha+\beta)}\|(f^{\mathrm{pre}} - f^{(i-1)})(t,x)\|^2 + \frac{\nabla v_t^\star(x)\{\alpha f^{\mathrm{pre}} + \beta f^{(i-1)}\}}{(\alpha+\beta)}.
\end{aligned}$$

Therefore, the HJB equation is reduced to

$$\begin{aligned}
&\frac{\sigma^2(t)}{2}\sum_{i,j}\frac{d^2 v_t^\star(x)}{dx^{[i]}dx^{[j]}} + \frac{dv_t^\star(x)}{dt} + \frac{\sigma^2(t)\|\nabla v_t^\star(x)\|_2^2}{2(\alpha+\beta)}\\
&- \frac{\alpha\beta}{2\sigma^2(t)(\alpha+\beta)}\|(f^{\mathrm{pre}} - f^{(i-1)})(t,x)\|^2 + \frac{\nabla v_t^\star(x)\{\alpha f^{\mathrm{pre}}(t,x) + \beta f^{(i-1)}(t,x)\}}{(\alpha+\beta)} = 0,
\end{aligned} \tag{16}$$

and the maximizer is

$$u^\star(t,x) = \sigma^2(t)\frac{\nabla_x v_t^\star(x)}{\alpha+\beta} + \frac{\alpha f^{\mathrm{pre}}(t,x) + \beta f^{(i-1)}(t,x)}{\alpha+\beta}.$$

Using the above and denoting $\gamma = \alpha + \beta$, we can further show

$$\begin{aligned}
&\frac{\sigma^2(t)}{2}\sum_{i,j}\frac{d^2\exp(v_t^\star(x)/\gamma)}{dx^{[i]}dx^{[j]}} + \frac{d\exp(v_t^\star(x)/\gamma)}{dt} - \frac{\alpha\beta\exp(v_t^\star(x)/\gamma)}{2\sigma^2(t)(\alpha+\beta)^2}\|(f^{\mathrm{pre}} - f^{(i-1)})(t,x)\|^2 \\
&+ \frac{\nabla\exp(v_t^\star(x)/\gamma)\{\alpha f^{\mathrm{pre}}(t,x) + \beta f^{(i-1)}(t,x)\}}{(\alpha+\beta)} \\
&= \frac{1}{\alpha+\beta}\exp\left(\frac{v_t^\star(x)}{\alpha+\beta}\right)\left\{\frac{\sigma^2(t)}{2}\sum_{i,j}\frac{d^2 v_t^\star(x)}{dx^{[i]}dx^{[j]}} + \frac{dv_t^\star(x)}{dt} + \frac{\sigma^2(t)\|\nabla v_t^\star(x)\|_2^2}{2(\alpha+\beta)}\right. \\
&\left. - \frac{\alpha\beta}{2\sigma^2(t)(\alpha+\beta)}\|(f^{\mathrm{pre}} - f^{(i-1)})(t,x)\|^2 + \frac{\nabla v_t^\star(x)\{\alpha f^{\mathrm{pre}}(t,x) + \beta f^{(i-1)}(t,x)\}}{(\alpha+\beta)}\right\} \qquad \text{(Use (16))} \\
&= 0.
\end{aligned}$$

Finally, using the Feynman–Kac formula, we obtain the form of the soft optimal value function:

$$\exp\left(\frac{v_t^\star(x_t)}{\alpha+\beta}\right) = \mathbb{E}_{\mathbb{P}^{\bar f}}\left[\exp\left(\frac{(\hat r^{(i)} + \hat g^{(i)})(x_T)}{\alpha+\beta} - \int_{t=s}^T \frac{\alpha\beta\|f^{\mathrm{pre}}(s,x_s) - f^{(i-1)}(s,x_s)\|_2^2}{2(\alpha+\beta)^2\sigma^2(s)}ds\right)|x_t\right],$$

$$dx_t = \bar f(t,x_t)dt + \sigma(t)dw_t, \quad \bar f(t,x_t) = \frac{\{\alpha f^{\mathrm{pre}} + \beta f^{(i-1)}\}(t,x_t)}{(\alpha+\beta)}, \quad \bar\nu = \frac{\alpha\nu_{\mathrm{pre}} + \beta\nu^{\mathrm{pre}}}{\alpha+\beta},$$

where the initial condition is

$$(\hat r^{(i)} + \hat g^{(i)})(x) = v_T^\star(x).$$

C.2. Proof of Theorem 4

We use induction. Suppose that the statement holds at $i-1$. This is indeed proven at $i=1$ in Uehara et al. (2024, Theorem 1). So, in the following, we assume

$$\mathbb{P}^{(i-1)}_{\cdot|T}(\tau|x_T) = \mathbb{P}^{\mathrm{pre}}_{\cdot|T}(\tau|x_T). \tag{17}$$

Under the above inductive assumption $i-1$, we first aim to prove the following:

$$\exp\left(\frac{v_t^\star(x_t)}{\alpha+\beta}\right) = \int \exp\left(\frac{(\hat r^{(i)} + \hat g^{(i)})(x_T)}{\alpha+\beta}\right)\left\{\frac{d\mathbb{P}^{f^{\mathrm{pre}}}(x_T|x_t)}{d\mu}\right\}^{\frac{\alpha}{\alpha+\beta}}\left\{\frac{d\mathbb{P}^{f^{(i-1)}}(x_T|x_t)}{d\mu}\right\}^{\frac{\beta}{\alpha+\beta}} d\mu \tag{18}$$

where μ is the Lebsgue measure.

Proof of (18). To prove it, we use the following:

$$1 = \int\left\{\frac{d\mathbb{P}^{f^{\mathrm{pre}}}(x_{[t,T]}|x_t,x_T)}{d\mathbb{P}^{\bar f}(x_{[t,T]}|x_t,x_T)}\right\}^{\frac{\alpha}{\alpha+\beta}}\left\{\frac{d\mathbb{P}^{f^{(i-1)}}(x_{[t,T]}|x_t,x_T)}{d\mathbb{P}^{\bar f}(x_{[t,T]}|x_t,x_T)}\right\}^{\frac{\beta}{\alpha+\beta}} \times d\mathbb{P}^{\bar f}(x_{[t,T]}|x_t,x_T). \tag{19}$$

This is proved by

$$\begin{aligned}
&\int\left\{\frac{d\mathbb{P}^{f^{\mathrm{pre}}}(x_{[t,T]}|x_t,x_T)}{d\mathbb{P}^{\bar f}(x_{[t,T]}|x_t,x_T)}\right\}^{\frac{\alpha}{\alpha+\beta}}\left\{\frac{d\mathbb{P}^{f^{(i-1)}}(x_{[t,T]}|x_t,x_T)}{d\mathbb{P}^{\bar f}(x_{[t,T]}|x_t,x_T)}\right\}^{\frac{\beta}{\alpha+\beta}} \times d\mathbb{P}^{\bar f}(x_{[t,T]}|x_t,x_T) \\
&= \int\left\{\frac{d\mathbb{P}^{f^{\mathrm{pre}}}(x_{[t,T]}|x_t,x_T)}{d\mathbb{P}^{\bar f}(x_{[t,T]}|x_t,x_T)}\right\} \times d\mathbb{P}^{\bar f}(x_{[t,T]}|x_t,x_T) = 1.
\end{aligned}$$

(Use $\frac{d\mathbb{P}^{f^{\mathrm{pre}}}(x_{[t,T]}|x_t,x_T)}{d\mathbb{P}^{f^{(i-1)}}(x_{[t,T]}|x_t,x_T)} = 1$ from the inductive assumption (17).)

Then, using (19), this is proved as follows:

$$
\begin{aligned}
(c) :=& \int \exp\left(\frac{(\hat{r}^{(i)}+\hat{g}^{(i)})(x_T)}{\alpha+\beta}\right)\left\{\frac{d\mathbb{P}^{f^{\mathrm{pre}}}(x_T|x_t)}{d\mu}\right\}^{\frac{\alpha}{\alpha+\beta}}\left\{\frac{d\mathbb{P}^{f^{(i-1)}}(x_T|x_t)}{d\mu}\right\}^{\frac{\beta}{\alpha+\beta}} d\mu \\
=& \int \exp\left(\frac{(\hat{r}^{(i)}+\hat{g}^{(i)})(x_T)}{\alpha+\beta}\right)\left\{\frac{d\mathbb{P}^{f^{\mathrm{pre}}}(x_T|x_t)}{d\mathbb{P}^{\bar{f}}(x_T|x_t)}\right\}^{\frac{\alpha}{\alpha+\beta}}\left\{\frac{d\mathbb{P}^{f^{(i-1)}}(x_T|x_t)}{d\mathbb{P}^{\bar{f}}(x_T|x_t)}\right\}^{\frac{\beta}{\alpha+\beta}} d\mathbb{P}^{\bar{f}}(x_T|x_t) \qquad \text{(Importance sampling)} \\
=& \int \exp\left(\frac{(\hat{r}^{(i)}+\hat{g}^{(i)})(x_T)}{\alpha+\beta}\right)\left\{\frac{d\mathbb{P}^{f^{\mathrm{pre}}}(x_{[t,T]}|x_t)}{d\mathbb{P}^{\bar{f}}(x_{[t,T]}|x_t)}\right\}^{\frac{\alpha}{\alpha+\beta}}\left\{\frac{d\mathbb{P}^{f^{(i-1)}}(x_{[t,T]}|x_t)}{d\mathbb{P}^{\bar{f}}(x_{[t,T]}|x_t)}\right\}^{\frac{\beta}{\alpha+\beta}} \{d\mathbb{P}^{\bar{f}}(x_T|x_t)\times d\mathbb{P}^{\bar{f}}(x_{[t,T]}|x_t,x_T)\}.
\end{aligned}
$$

Here, from the second line to the third line, we use (19), noticing

$$
\frac{d\mathbb{P}^{f^{\mathrm{pre}}}(x_{[t,T]}|x_t)}{d\mathbb{P}^{\bar{f}}(x_{[t,T]}|x_t)} = \frac{d\{\mathbb{P}^{f^{\mathrm{pre}}}(x_{[t,T]}|x_t,x_T)\mathbb{P}^{f^{\mathrm{pre}}}(x_T|x_t)\}}{d\{\mathbb{P}^{\bar{f}}(x_{[t,T]}|x_t,x_T)\mathbb{P}^{\bar{f}}(x_T|x_t)\}},\ \frac{d\mathbb{P}^{f^{(i-1)}}(x_{[t,T]}|x_t)}{d\mathbb{P}^{\bar{f}}(x_{[t,T]}|x_t)} = \frac{d\{\mathbb{P}^{f^{(i-1)}}(x_{[t,T]}|x_t,x_T)\mathbb{P}^{f^{(i-1)}}(x_T|x_t)\}}{d\{\mathbb{P}^{\bar{f}}(x_{[t,T]}|x_t,x_T)\mathbb{P}^{\bar{f}}(x_T|x_t)\}}.
$$

Finally, we have

$$
\begin{aligned}
(c) =& \int \exp\left(\frac{(\hat{r}^{(i)}+\hat{g}^{(i)})(x_T)}{\alpha+\beta}\right)\left\{\frac{d\mathbb{P}^{f^{\mathrm{pre}}}(x_{[t,T]}|x_t)}{d\mathbb{P}^{\bar{f}}(x_{[t,T]}|x_t)}\right\}^{\frac{\alpha}{\alpha+\beta}}\left\{\frac{d\mathbb{P}^{f^{(i-1)}}(x_{[t,T]}|x_t)}{d\mathbb{P}^{\bar{f}}(x_{[t,T]}|x_t)}\right\}^{\frac{\beta}{\alpha+\beta}} d\mathbb{P}^{\bar{f}}(x_{[t,T]}|x_t) \\
=& \int \exp\left(\frac{(\hat{r}^{(i)}+\hat{g}^{(i)})(x_T)}{\alpha+\beta}\right)\exp\left(-\int_{s=t}^{T}\frac{\alpha\beta\|(f^{\mathrm{pre}}(s,x_s)-f^{(i-1)})(s,x_s)\|_2^2}{2(\alpha+\beta)^2\sigma^2(s)}ds\right) d\mathbb{P}^{\bar{f}}(x_{[t,T]}|x_t) \\
=& \mathbb{E}_{\mathbb{P}^{\bar{f}}}\left[\exp\left(\frac{(\hat{r}^{(i)}+\hat{g}^{(i)})(x_T)}{\alpha+\beta}-\alpha\beta\int_{s=t}^{T}\frac{\|(f^{(i-1)}-f^{\mathrm{pre}})(s,x_s)\|^2}{2\sigma^2(s)}ds\right)|x_t=x\right].
\end{aligned}
$$

From the first line to the second line, we use the Girsanov theorem:

$$
\begin{aligned}
\left\{\frac{d\mathbb{P}^{f^{\mathrm{pre}}}(x_{(t,T]}|x_t)}{d\mathbb{P}^{\bar{f}}(x_{(t,T]}|x_t)}\right\}^{\frac{\alpha}{\alpha+\beta}} &= \exp\left(\frac{\alpha}{\alpha+\beta}\left\{\beta\int_{s=t}^{T}\frac{(f^{\mathrm{pre}}-f^{(i-1)})(s,x_s)}{\sigma(s)}dw_s-\int_{s=t}^{T}\frac{\alpha\beta\|(f^{\mathrm{pre}}-f^{(i-1)})(s,x_s)\|_2^2}{2(\alpha+\beta)^2\sigma^2(s)}ds\right\}\right), \\
\left\{\frac{d\mathbb{P}^{f^{(i-1)}}(x_{(t,T]}|x_t)}{d\mathbb{P}^{\bar{f}}(x_{(t,T]}|x_t)}\right\}^{\frac{\beta}{\alpha+\beta}} &= \exp\left(\frac{\beta}{\alpha+\beta}\left\{\alpha\int_{s=t}^{T}\frac{(f^{(i-1)}-f^{\mathrm{pre}})(s,x_s)}{\sigma(s)}dw_s-\int_{s=t}^{T}\frac{\alpha\beta\|(f^{\mathrm{pre}}-f^{(i-1)})(s,x_s)\|_2^2}{2(\alpha+\beta)^2\sigma^2(s)}ds\right\}\right).
\end{aligned}
$$

Hence, using (15) in Theorem 3, we can conclude

$$
(c) = \exp\left(\frac{v_t^\star(x_t)}{\alpha+\beta}\right).
$$

Main part of the proof. Now, we aim to prove the optimal distribution over trajectory $(\mathcal{C})$ is

$$
\frac{1}{C}\exp\left(\frac{\hat{r}^{(i)}(x_T)+\hat{g}^{(i)}(x_T)}{\alpha+\beta}\right)\{\mathbb{P}^{(i-1)}(\tau)\}^{\frac{\beta}{\alpha+\beta}}\{\mathbb{P}^{\mathrm{pre}}(\tau)\}^{\frac{\alpha}{\alpha+\beta}}.
$$

To achieve this, we first show that the conditional optimal distribution over $\mathcal{C}$ at state x_0 (i.e., $\mathbb{P}^{(i-1)}_{\cdot|0}(\tau|x_0)$) is

$$
\frac{1}{\exp\left(\frac{v_0^\star(x_0)}{\alpha+\beta}\right)}\times\exp\left(\frac{\hat{r}^{(i)}(x_T)+\hat{g}^{(i)}(x_T)}{\alpha+\beta}\right)\{\mathbb{P}^{(i-1)}(\tau|x_0)\}^{\frac{\beta}{\alpha+\beta}}\{\mathbb{P}^{\mathrm{pre}}(\tau|x_0)\}^{\frac{\alpha}{\alpha+\beta}}. \tag{20}
$$

First, we need to check that this is a valid distribution over $\mathcal{C}$. This is because using an inductive hypothesis:

$$
\mathbb{P}^{\mathrm{pre}}(\tau|x_T,x_0) = \mathbb{P}^{(i-1)}(\tau|x_T,x_0),
$$

which is clear from (17), the above can be decomposed into

$$\underbrace{\left[\frac{1}{\exp\left(\frac{v_0^\star(x_0)}{\alpha+\beta}\right)} \times \exp\left(\frac{\hat{r}^{(i)}(x_T)+\hat{g}^{(i)}(x_T)}{\alpha+\beta}\right)\{\mathbb{P}^{(i-1)}(x_T|x_0)\}^{\frac{\beta}{\alpha+\beta}}\{\mathbb{P}^{\text{pre}}(x_T|x_0)\}^{\frac{\alpha}{\alpha+\beta}}\right]}_{(\alpha 1)} \times \underbrace{\mathbb{P}^{\text{pre}}(\tau|x_T,x_0)}_{(\alpha 2)}.$$

Here, note that both term $(\alpha 1)$, $(\alpha 2)$ are normalized. Especially, to check $(\alpha 1)$ is normalized, we can use Equation (18) at $t=0$.

Now, after checking that (20) is a valid distribution, we calculate the KL divergence. This is calculated as follows:

$$\begin{aligned}
&\text{KL}\left(\mathbb{P}^{(i)}_{\cdot|0}(\tau|x_0)\|\frac{1}{\exp\left(\frac{v_0^\star(x_0)}{\alpha+\beta}\right)\nu^{\text{pre}}(x_0)} \times \exp\left(\frac{\hat{r}^{(i)}(x_T)+\hat{g}^{(i)}(x_T)}{\alpha+\beta}\right)\{\mathbb{P}^{(i-1)}(\tau)\}^{\frac{\beta}{\alpha+\beta}}\{\mathbb{P}^{\text{pre}}(\tau)\}^{\frac{\alpha}{\alpha+\beta}}\right)\\
&= \text{KL}\left(\mathbb{P}^{(i)}_{\cdot|0}(\tau|x_0)\|\{\mathbb{P}^{(i-1)}_{\cdot|0}(\tau|x_0)\}^{\frac{\beta}{\alpha+\beta}}\{\mathbb{P}^{\text{pre}}(\tau|x_0)\}^{\frac{\alpha}{\alpha+\beta}}\right) + \mathbb{E}_{\tau\sim\mathbb{P}^{(i)}}\left[\left(\frac{\hat{r}^{(i)}(x_T)+\hat{g}^{(i)}(x_T)}{\alpha+\beta}\right)\right] - \frac{v_0^\star(x_0)}{\alpha+\beta}\\
&= \mathbb{E}_{\tau\sim\mathbb{P}^{(i)}}\left[\left(\frac{\hat{r}^{(i)}(x_T)+\hat{g}^{(i)}(x_T)}{\alpha+\beta}\right) - \int_{s=0}^{T}\frac{\beta\|(f^{(i)}-f^{(i-1)})(s,x_s)\|_2^2+\alpha\|(f^{\text{pre}}-f^{(i)})(s,x_s)\|_2^2}{2(\alpha+\beta)\sigma^2(s)}ds\right] - \frac{v_0^\star(x_0)}{\alpha+\beta}\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\text{(Use Girsanov theorem)}\\
&= 0 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\text{(Use Definition of soft optimal value functions)}
\end{aligned}$$

Hence, we can now conclude (20).

Next, we consider an exact formulation of the optimal initial distribution. We just need to solve

$$\underset{\nu\in\Delta(\mathcal{X})}{\text{argmax}}\int v_0^\star(x)\nu(x)dx - \beta\text{KL}(\nu\|\nu^{(i-1)}) - \alpha\text{KL}(\nu\|\nu^{\text{pre}}).$$

The closed-form solution is proportional to

$$\frac{1}{C}\exp\left(\frac{v_0^\star(x)}{\alpha+\beta}\right)\{\nu^{(i-1)}(x)\}^{\frac{\alpha}{\alpha+\beta}}\nu^{\text{pre}}(x)\}^{\frac{\alpha}{\alpha+\beta}}. \tag{21}$$

Finally, by multiplying (21) and (20), we can conclude that the optimal distribution over $\mathcal{C}$ is

$$\frac{1}{C}\exp\left(\frac{\hat{r}^{(i)}(x_T)+\hat{g}^{(i)}(x_T)}{\alpha+\beta}\right)\{\mathbb{P}^{(i-1)}(\tau)\}^{\frac{\beta}{\alpha+\beta}}\{\mathbb{P}^{\text{pre}}(\tau)\}^{\frac{\alpha}{\alpha+\beta}}.$$

D. Experiment Details

D.1. Implementation of Baselines

In this section, we describe more details of the baselines.

Online PPO. Considering the discretized formulation of diffusion models (Black et al., 2023; Fan et al., 2023), we use the following update rule:

$$\nabla_\theta\mathbb{E}_{\mathcal{D}}\sum_{t=1}^{T}\left[\min\left\{\tilde{r}_t(x_0,x_t)\frac{p(x_t|x_{t-1};\theta)}{p(x_t|x_{t-1};\theta_{\text{old}})},\tilde{r}_t(x_0,x_t)\cdot\text{Clip}\left(\frac{p(x_t|x_{t-1};\theta)}{p(x_t|x_{t-1};\theta_{\text{old}})},1-\epsilon,1+\epsilon\right)\right\}\right], \tag{22}$$

$$\tilde{r}_t(x_0,x_t) = -r(x_T) + \underbrace{\alpha\frac{\|u(t,x_t;\theta)\|^2}{2\sigma^2(t)}}_{\text{KL term}}, \quad p(x_t|x_{t-1};\theta) = \mathcal{N}(u(t,x_t;\theta)+f(t,x_t),\sigma(t)) \tag{23}$$

Here, note a pre-trained diffusion model is denoted by $p(x_t|x_{t-1};\theta) = \mathcal{N}(f(t, x_{t-1}), \sigma)$ and θ is a parameter to be optimized.

Note that DPOK (Fan et al., 2023) uses the following update:

$$\nabla_\theta \mathbb{E}_{\mathcal{D}} \sum_{t=1}^{T} \left[\min\left\{ -r(x_0)\frac{p(x_t|x_{t-1};\theta)}{p(x_t|x_{t-1};\theta_{\text{old}})}, -r(x_0)\cdot \text{Clip}\left(\frac{p(x_t|x_{t-1};\theta)}{p(x_t|x_{t-1};\theta_{\text{old}})}, 1-\epsilon, 1+\epsilon\right)\right\} + \underbrace{\alpha\frac{\|u(t,x_t;\theta)\|^2}{2\sigma^2(t)}}_{\text{KL term}}\right]$$

where the KL term is directly differentiated. We did not use the DPOK update rule because DDPO appears to outperform DPOK even without a KL penalty (Black et al. (2023), Appendix C), so we implemented this baseline by modifying the DDPO codebase to include the added KL penalty term (Equation (23)).

Guidance. We use the following implementation of guidance (Dhariwal and Nichol, 2021):

- For each $t \in [0, T]$, we train a model: $p(y|x_t)$ where x_t is a random variable induced by the pre-trained diffusion model.
- We fix a guidance level $\gamma \in \mathbb{R}_{>0}$, target value $y_{\text{con}} \in \mathbb{R}$, and at inference time (during each sampling step), we use the following score function

$$\nabla_x \log p(x|y = y_{\text{con}}) = \nabla_x \log p(x) + \gamma \nabla_x \log p(y = y_{\text{con}}|x).$$

A remaining question is how to model $p(y|x)$. In our case, for the biological example, we make a label depending on whether x is top 10% or not and train a binary classifier. In image experiments, we construct a Gaussian model: $p(y|x) = \mathcal{N}(y - \mu_\theta(x), \sigma^2)$ where y is the reward label, μ_θ is the reward model we need to train, and σ is a fixed hyperparameter.

D.2. Experiment in Molecules

We do an analogous experiment in Section 7.1 to generate molecules with improved properties, specifically the Quantitative Estimate of Druglikeness (QED) score. It's important to note that this experiment is conducted on a simplified scale, as we utilize a trained oracle instead of a real black box oracle for evaluation.

This focuses on obtaining a module with favorable properties. Here, x represents the molecule, and y corresponds to the Quantitative Estimate of Druglikeness (QED) score. We employ a pre-trained diffusion model using the ZINC dataset (Irwin and Shoichet, 2005). In this case, we use a graph neural network-based diffusion (Jo et al., 2022). To quantify diversity, we employ the $(1 -$ Tanimoto coefficient$)$ of the standard molecular fingerprints (Jo et al., 2022; Bajusz et al., 2015). We set $M_i = 2500$ and $K = 4$ ($M = 10000$).

The results are reported in Table 4 on page 22 and Figure 3. We observe similar trends in Section 7.1.

Table 4. Results for fine-tuning diffusion models for molecules using the ZINC dataset to optimize QED scores. **SEIKO** attains high rewards using a fixed budget of feedback.

	Value ↑	Div ↑
Non-adaptive	0.82 ± 0.02	0.85
Guidance	0.73 ± 0.02	**0.88**
Online PPO	0.76 ± 0.03	0.85
Greedy	0.72 ± 0.00	0.74
UCB (Ours)	0.86 ± 0.01	0.86
Bootstrap (Ours)	$\mathbf{0.88 \pm 0.01}$	0.83

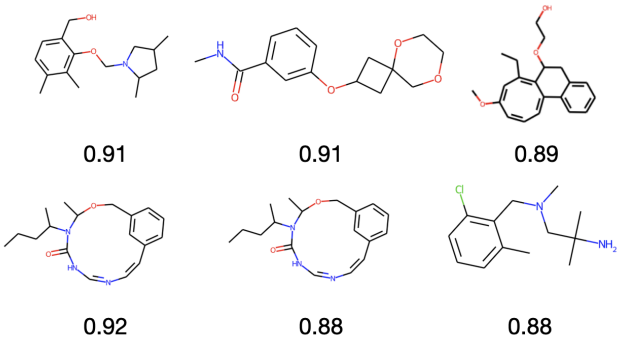

Figure 3. Examples of molecules that attain high QED generated by **SEIKO**.

Table 5. Architecture of diffusion models for GFP

Layer	Input Dimension	Output dimension	Explanation
1	$1\ (t)$	$256\ (t')$	Get time feature
1	$237 \times 20\ (x)$	$64\ (x')$	Get positional encoder (Denote x')
2	$237 \times 20 + 256 + 64\ (x, t, x')$	$64\ (\bar{x})$	Transformer Encoder
3	$64\ (\bar{x})$	$237 \times 20\ (x)$	Linear

D.3. Details for Protein Sequences and Molecules

D.3.1. Description of Data

GFP. The original dataset size is 56086. Each data consists of an amino acid sequence with 8-length. We represent each data as a one-hot encoding representation with dimension 237×20. Here, we model the difference between the original and baseline sequences. We selected the top 33637 samples following Trabucco et al. (2022) and trained diffusion models and oracles using this selected data.

ZINC. The ZINC dataset for molecules is a large and freely accessible collection of chemical compounds used in drug discovery and computational chemistry research (Irwin and Shoichet, 2005). The dataset contains a diverse and extensive collection of chemical structures, including small organic molecules.

The dataset size we use is $249,455$. We use the entire dataset to acquire pre-trained diffusion models. For the numerical representation of each molecule, we adopt a graph representation denoted as $x \in \mathbb{R}^{38\times 47}$. This representation comprises two matrices: $x_1 \in \mathbb{R}^{38\times 38}$ and $x_2 \in \mathbb{R}^{38\times 9}$. A matrix x_1 serves as an adjacency matrix for the molecule, while x_2 encodes features for each node. With 38 nodes, in total, each node is associated with a 9-dimensional feature vector representing its correspondence to a specific element.

D.3.2. Architecture of Neural Networks

Concerning diffusion models and fine-tuning for ZINC, we adopt the architecture presented in Jo et al. (2022), specifically designed for graph representation. They have implemented this architecture by effectively stacking transformed and GFN components. Therefore, we will not delve into the details here.

Regarding diffusion models and fine-tuning for GFP, we add detail for protein sequences in Table 5, 6.

D.3.3. Hyperparameters

In all our implementations, we utilize A100 GPUs. For the fine-tuning of diffusion models, we employ the specific set of hyperparameters outlined in Table 7.

Table 6. Architecture of oracles for GFP

	Input dimension	Output dimension	Explanation
1	237×20	500	Linear
1	500	500	ReLU
2	500	200	Linear
2	200	200	ReLU
3	200	1	Linear
3	200	1	ReLU
4	1	1	Sigmoid

Table 7. Important hyperparameters for fine-tuning. For all methods, we use ADAM as an optimizer.

Method	Type	GFP	ZINC
SEIKO	Batch size	128	32
	KL parameter β	0.01	0.002
	UCB parameter C_1	0.002	0.002
	Number of bootstrap heads	3	3
	Sampling to neural SDE	Euler	Maruyama
	Step size (fine-tuning)	50	25
	Epochs (fine-tuning)	100	100
PPO	Batch size	128	128
	ϵ	0.1	0.1
	Epochs	100	100
Guidance	Guidance level	10	10
Pre-trained diffusion	Forward SDE	Variance preserving	
	Sampling way	Euler	Maruyama

D.4. Details for Image Tasks

Prompts. Since StableDiffusion is a text-to-image model, we need to specify prompts used for training and evaluation, respectively. To align with prior studies, our training process involves using prompts from a predefined list of 50 animals (Black et al., 2023; Prabhudesai et al., 2023). In evaluation, we adopt the following animals: snail, hippopotamus, cheetah, crocodile, lobster, and octopus; all are not seen during training.

Techniques for saving CUDA memory. We use the DDIM sampler (Song et al., 2020) with 50 steps for sampling. To manage memory constraints during back-propagation through the sampling process and the VAE decoder, we implemented two solutions from Clark et al. (2023); Prabhudesai et al. (2023): (1) Fine-tuning LoRA modules (Hu et al., 2021) instead of the full diffusion weights, and (2) Using gradient checkpointing (Gruslys et al., 2016; Chen et al., 2016). We also applied randomized truncated back-propagation, limiting gradient back-propagation to a random number of steps denoted as K. In practice, K is uniformly obtained from $(0, 50)$ as Prabhudesai et al. (2023).

Guidance. To approximate the classifier $p(y|x_t)$, we train a reward model $\mu_\theta(x_t, t)$ on the top of the OpenAI CLIP embeddings (Radford et al., 2021). The reward model is implemented as an MDP, which inputs the concatenation of sinusoidal time embeddings (for time t) and CLIP embeddings (for x_t). Our implementation is based on the public RCGDM (Yuan et al., 2023) codebase[1].

Online PPO. We implement a PPO-based algorithm, specifically DPOK (Fan et al., 2023), configured with $M = 15360$ feedback interactions. The implementation is based on the public DDPO (Black et al., 2023) codebase[2].

[1] https://github.com/Kaffaljidhmah2/RCGDM
[2] https://github.com/kvablack/ddpo-pytorch

Table 8. Important hyperparameters for fine-tuning Aesthetic Scores.

SEIKO	Batch size per GPU	2
	Samples per iteration	64
	Samples per epoch	128
	KL parameter β	1
	UCB parameter C_1	0.01
	UCB parameter λ	0.001
	Number of bootstrap heads	4
	DDIM Steps	50
	Guidance weight	7.5
PPO	Batch size per GPU	4
	Samples per iteration	128
	Samples per epoch	256
	KL parameter β	0.001
	ϵ	1e-4
	Epochs	60
Guidance	Guidance level	400
Optimization	Optimizer	AdamW
	Learning rate	1e-4
	(ϵ_1, ϵ_2)	$(0.9, 0.999)$
	Weight decay	0.1
	Clip grad norm	5
	Truncated back-propagation step K	$K \sim \text{Uniform}(0, 50)$

D.4.1. Hyperparameters

In all image experiments, we use four A100 GPUs for fine-tuning StableDiffusion v1.5 (Rombach et al., 2022). Full training hyperparameters are listed in Table 8

D.4.2. Additional Results

Training Curves. We plot the training curves in Figure 4. Recall we perform a four-shot online fine-tuning process. For each online iteration, the model is trained with 5 epochs. In Table 3, we report the final results after 20 epochs.

Additional Generated Images. In Figure 5, we provide more qualitative samples to illustrate the performance of **SEIKO** in fine-tuning aesthetic quality.

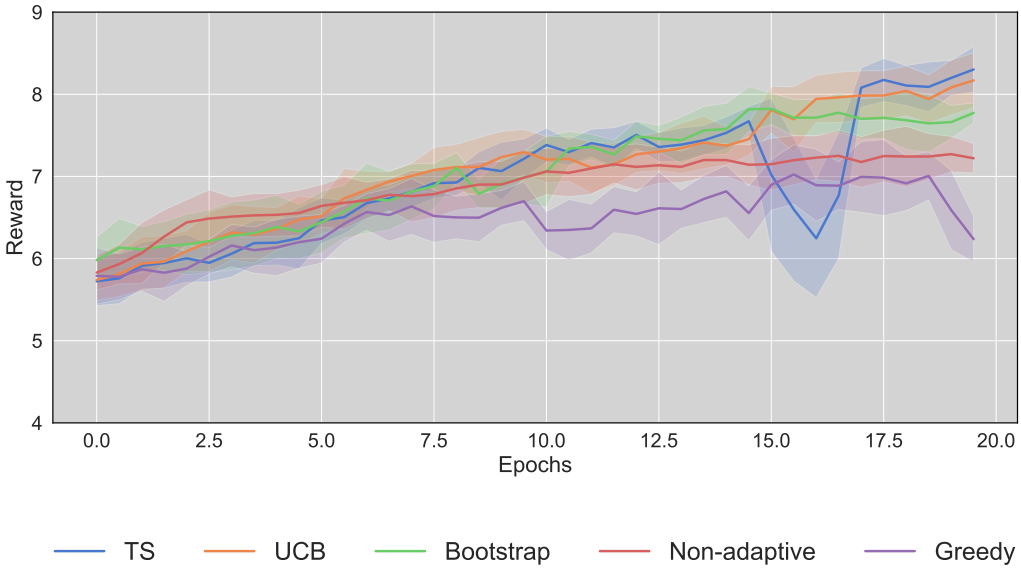

Figure 4. Training curves of reward $r(x)$ for fine-tuning aesthetic scores.

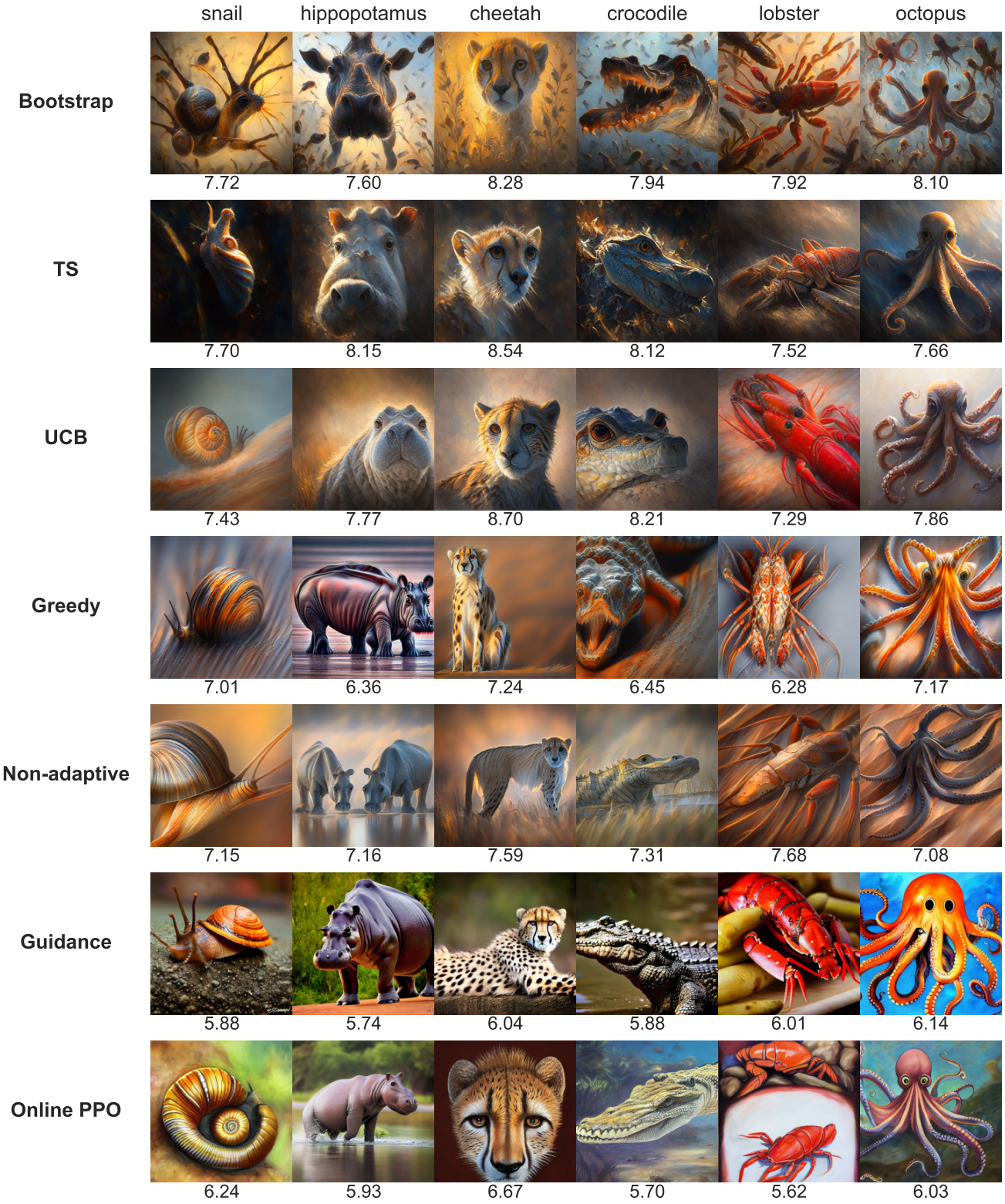

Figure 5. More images generated by **SEIKO** and baselines. For all algorithms, fine-tuning is conducted with a total of 15600 samples.