DIFFUSION POSTERIOR SAMPLING FOR FAST ADAPTATION IN MULTI-TASK NONLINEAR CONTEXTUAL BANDITS

Anonymous authors
Paper under double-blind review

ABSTRACT

We study fast adaptation in multi-task nonlinear contextual bandits, where different tasks share the same reward structure but are characterized by distinct model parameters drawn from a common unknown prior distribution. Given data from past tasks, the goal is to rapidly adapt to a new task and minimize regret using only limited online interactions. Thompson Sampling (TS) is a popular approach for solving contextual bandits, maintaining a posterior over the model parameter that is updated each round using a hand-specified conjugate prior (e.g., Gaussian) and the observed rewards. However, such priors cannot capture the rich cross-task structure in multi-task settings, leading to misspecified posteriors and suboptimal exploration. To enable fast adaptation, we train a diffusion model on data from past tasks to learn a flexible transferable prior distribution over task parameters. In a new bandit task, parameters are estimated via a conditional reverse-diffusion process, where each step combines: (i) an unconditional drift from the diffusion prior, (ii) a likelihood-driven drift from the interaction history, and (iii) a noise term enabling randomized exploration. We instantiate this framework in two ways. **DLTS** integrates history into the diffusion prior at every reverse step to form a conditional posterior, from which approximate samples are drawn. **DPSG** applies a single history-guided gradient correction at each diffusion step. Theoretically, we formalize oracle TS and its diffusion counterpart, and show they are equivalent when the diffusion prior matches the true prior. Empirically, our methods are competitive with specialized baselines in linear settings and provide strong fast-adaptation performance in challenging nonlinear bandit environments, requiring substantially fewer online interactions to reach a target performance level.

1 INTRODUCTION

Sequential decision-making under uncertainty hinges on balancing exploration and exploitation (Lattimore & Szepesvári, 2020). One prominent approach to address this trade-off is Thompson Sampling (TS) (Thompson, 1933). In contextual bandits, TS maintains a posterior over the model parameter that is updated each round using a hand-specified conjugate prior (e.g., Gaussian) and the observed rewards (Agrawal & Goyal, 2013). In multi-task settings where different tasks share the same reward structure but are characterized by distinct model parameters drawn from a common unknown prior distribution, such simple priors fail to capture rich cross-task structure, leading to misspecified posteriors and suboptimal exploration (Chapelle & Li, 2011). To address this, Hong et al. (2022) proposed to initialize TS with a mixture prior for multi-task bandits. Nevertheless, fixed parametric mixtures still struggle with complex structure—e.g., multimodal task families where the parameter distribution has many distinct peaks (modes) that can be uneven in size or far apart (Vuorio et al., 2019; Finn et al., 2018), and heavy-tailed families with extreme outliers (Bubeck et al., 2013; Forbes & Wraith, 2014). This motivates learning a flexible prior that fully leverages past tasks to enable fast adaptation on a new task with limited interaction.

Deep generative models have achieved remarkable success in producing high-quality synthetic data across modalities (Saharia et al., 2022; Rombach et al., 2022; Liu et al., 2023). These results highlight their ability to model complex, multi-modal distributions. In online decision-making, the policy must incorporate newly collected interaction data and update its strategy frequently. Diffusion

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

models (Sohl-Dickstein et al., 2015; Ho et al., 2020) align well with this need: their reverse process is iterative and naturally supports conditioning on new observations at every round. In score-based diffusion (Song et al., 2021), this conditioning is implemented by augmenting the score with a likelihood term (Chung et al., 2023). Given this intuition, prior work (Hsieh et al., 2023) tackles multi-task multi-armed bandits by training a diffusion model to learn a flexible prior over task parameters and coupling it with Thompson Sampling, with practical steps such as variance calibration to obtain reliable uncertainty in the reverse process (Hsieh et al., 2023). In parallel, diffusion priors have been adapted to online posterior sampling for linear and generalized linear contextual bandits via closed-form updates at each reverse step (Kveton et al., 2024). However, this prior work focuses on specialized updates for linear or generalized linear bandits, leaving multi-task nonlinear contextual bandits largely unexplored. The core difficulty is the absence of closed-form diffusion reverse updates for nonlinear reward models, which complicates algorithm design. Without closed forms, sampling error can be large; combined with limited interactions per task and potential inaccuracies in the diffusion learned prior, this can destabilize posterior sampling and misguide exploration. Consequently, a natural question arises:

Can we learn a flexible diffusion prior from historical tasks and use it for posterior sampling that enables fast adaptation in general nonlinear contextual bandits?

In this paper, we provide an affirmative answer to this question by developing a diffusion-based posterior sampling framework for fast adaptation in multi-task nonlinear contextual bandits. We learn a flexible prior over task parameters with a diffusion model trained on past tasks, and on a new task perform conditional reverse sampling that blends prior drift, likelihood guidance, and stochastic exploration at each step. We instantiate this with two methods: DLTS, which conditions at every reverse step and samples via Langevin Monte Carlo (LMC), and DPSG, which draws an unconditional sample and applies a single likelihood-guided correction at every reverse step. To stabilize learning, we also propose DPSG-MP, a practical DPSG variant that replaces the single correction with a short inner loop of likelihood-gradient updates with projection. **Our contributions** are summarized as follows:

- **Unified Framework.** We present a unified diffusion-based posterior sampling framework for fast adaptation in multi-task nonlinear contextual bandits. In our framework, the algorithm estimates the reward model parameters via a conditional reverse-diffusion process that, at each step, consists of (i) an unconditional drift from the diffusion prior, which transfers cross-task structure from past tasks, (ii) a likelihood-driven drift from the interaction history, which adapts the sample to the current task, and (iii) a noise term enabling randomized exploration.
- **Algorithm Design.** We instantiate the unified framework in two ways: (i) Diffusion Langevin Thompson Sampling (DLTS), which explicitly constructs the conditional posterior and then draws approximate samples; and (ii) Diffusion Posterior Sampling with Guidance (DPSG), which couples unconditional sampling with a single history-guided correction step for simplicity and speed. We also propose DPSG with Multi-step Projection (DPSG-MP), a practical DPSG variant to stabilize learning, which improves empirical performance.
- **Theoretical analysis.** We formalize oracle TS (OTS), oracle diffusion TS (ODTS) and prove they are equivalent when the diffusion prior exactly matches the true prior.
- **Extensive experiments.** We evaluate our algorithms in simulations of both linear and non-linear contextual bandits. In the linear setting, our algorithms achieve performance comparable to baseline algorithms that use closed-form updates, such as LinTS and DiffTS. Furthermore, our methods demonstrate strong performance in the non-linear bandit setting, where the learned diffusion prior provides a significant advantage. Our wall-clock time analysis also shows strong computational efficiency compared with baselines.

2 PRELIMINARIES

Nonlinear Contextual Bandits. Contextual bandits form a broad class of sequential decision problems where the player chooses based on an observed action set represented by feature vectors (contexts). At each round t, the player observes arm set $\mathcal{X}_t \subseteq \mathbb{R}^d$, selects an arm $\boldsymbol{x}_t \in \mathcal{X}_t$ and receives reward y_t from the environment. Assume that the mean reward for a feature $\boldsymbol{x} \in \mathbb{R}^d$ is generated by an underlying function $f(\boldsymbol{\theta}^*; \boldsymbol{x})$, and the observed reward satisfies $y(\boldsymbol{x}) = f(\boldsymbol{\theta}^*; \boldsymbol{x}) + \varepsilon$, where $\boldsymbol{\theta}^* \in \mathbb{R}^d$ is an unknown parameter shared by all arms and ε is observation noise. The goal of

a bandit algorithm is to maximize cumulative reward over a horizon T, equivalently to minimize the pseudo-regret (Lattimore & Szepesvári, 2020): $R(T) = \sum_{t=1}^{T}(f(\boldsymbol{\theta}^*; \boldsymbol{x}_t^\star) - f(\boldsymbol{\theta}^*; \boldsymbol{x}_t))$, where $\boldsymbol{x}_t^* = \arg\max_{\boldsymbol{x}\in\mathcal{X}_t} f(\boldsymbol{\theta}^*; \boldsymbol{x})$ is the arm with the highest expected reward at round t.

Multi-task Bandit Problem. All the bandit tasks share the same reward model structure $y(\boldsymbol{x}) = f(\boldsymbol{\theta}^*; \boldsymbol{x}) + \varepsilon$ and the underlying parameter $\boldsymbol{\theta}^*$ varies across tasks but is drawn independently from a common prior distribution $p_0(\boldsymbol{\theta}^*)$. In our problem setup, we have model parameters $\{\boldsymbol{\theta}^{(i)}\}_{i=1}^{N}$ from previous N bandit tasks, which can be ground-truth (perfect data) or estimation (imperfect data). Our goal is to utilize these data to help solve for a new bandit task $(N+1)$.

Diffusion Models. Score-based diffusion models define a generative process as the reverse of a continuous-time noising process that gradually perturbs data toward a tractable reference distribution. Let $\boldsymbol{\theta}_s \in \mathbb{R}^d$ denote the state at continuous time $s \in [0, S]$, with $\boldsymbol{\theta}_0 \sim p_0$ and $\boldsymbol{\theta}_S \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. In the variance-preserving (VP) formulation (Song et al., 2021), the forward SDE is: $\mathrm{d}\boldsymbol{\theta}_s = -\frac{1}{2}\beta_s\boldsymbol{\theta}_s\mathrm{d}s + \sqrt{\beta_s}\mathrm{d}\boldsymbol{w}_s$, where $\beta_s > 0$ is a noise schedule and $\boldsymbol{w}_s$ is a standard d-dimensional Wiener process. The corresponding reverse-time diffusion transports $\boldsymbol{\theta}_S \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ back to p_0. The reverse diffusion SDE is

$$\mathrm{d}\boldsymbol{\theta}_s = [-\tfrac{1}{2}\beta_s\boldsymbol{\theta}_s - \beta_s\nabla_{\boldsymbol{\theta}}\log p_s(\boldsymbol{\theta}_s)]\mathrm{d}s + \sqrt{\beta_s}\mathrm{d}\bar{\boldsymbol{w}}_s,$$

where $\bar{\boldsymbol{w}}_s$ is a standard Wiener process in reverse time and $\nabla_{\boldsymbol{\theta}}\log p_s(\boldsymbol{\theta}_s)$ is the time-dependent score function where $p_s(\boldsymbol{\theta}_s) = \int p_0(\boldsymbol{\theta}_0)p(\boldsymbol{\theta}_s|\boldsymbol{\theta}_0)\mathrm{d}\boldsymbol{\theta}_0$. Since score function $\nabla_{\boldsymbol{\theta}}\log p_s(\boldsymbol{\theta}_s)$ is unknown, a neural network $s_{\boldsymbol{\psi}}(\boldsymbol{\theta}_s, s)$ is trained to approximate it via denoising score matching:

$$\boldsymbol{\psi}^* = \arg\min_{\boldsymbol{\psi}} \mathbb{E}\|s_{\boldsymbol{\psi}}(\boldsymbol{\theta}_s, s) - \nabla_{\boldsymbol{\theta}}\log p\big(\boldsymbol{\theta}_s|\boldsymbol{\theta}_0\big)\|_2^2,$$

where in expectation $s \sim \mathcal{U}(\epsilon, 1), \boldsymbol{\theta}_0 \sim p_0, \boldsymbol{\theta}_s \sim p(\boldsymbol{\theta}_s|\boldsymbol{\theta}_0)$, $\epsilon > 0$ is a small constant and $p(\boldsymbol{\theta}_s|\boldsymbol{\theta}_0)$ is Gaussian. Plugging $s_{\boldsymbol{\psi}^*}$ into the reverse SDE and discretizing yields a sampler that approximately draws from p_0.

To obtain a practical training and sampling procedure, we discretize the time index into L noise levels indexed by $\ell \in \{1, \ldots, L\}$ and adopt discrete VP formulation. We define a discrete noise schedule $\beta_\ell > 0$, and define $\alpha_\ell = 1 - \beta_\ell, \bar{\alpha}_\ell = \prod_{j=1}^{\ell}\alpha_j$. The forward process has the closed form: $\boldsymbol{\theta}_\ell = \sqrt{\bar{\alpha}_\ell}\boldsymbol{\theta}_0 + \sqrt{1-\bar{\alpha}_\ell}\boldsymbol{z}$ where $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The reverse process involves iteratively sampling $\boldsymbol{\theta}_{\ell-1}$ from the posterior distribution $p(\boldsymbol{\theta}_{\ell-1}|\boldsymbol{\theta}_\ell)$. Instead of directly approximating the score function $\nabla_{\boldsymbol{\theta}_\ell}\log p_s(\boldsymbol{\theta}_\ell)$, it is common practice to reparameterize the model to predict the added noise $\boldsymbol{z}$ at step ℓ by denoiser network $\varepsilon_{\boldsymbol{\phi}^*}$. The reverse update then becomes an ancestral Gaussian sampling step: $\boldsymbol{\theta}_{\ell-1} \sim \mathcal{N}(\boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)$, where $\boldsymbol{\mu}_\ell = \frac{1}{\sqrt{\alpha_\ell}}(\boldsymbol{\theta}_\ell - \frac{1-\alpha_\ell}{\sqrt{1-\bar{\alpha}_\ell}}\varepsilon_{\boldsymbol{\phi}^*}(\boldsymbol{\theta}_\ell, \ell))$, $\boldsymbol{\Sigma}_\ell = \frac{1-\bar{\alpha}_{\ell-1}}{1-\bar{\alpha}_\ell}\beta_\ell\mathbf{I}$. This formulation is equivalent to DDPM (Ho et al., 2020) and serves as backbone for our diffusion prior.

3 Diffusion Posterior Sampling Framework

In this section, we introduce a diffusion posterior sampling framework for multi-task nonlinear contextual bandits, beginning with a unified diffusion reverse update and then two instantiations.

3.1 A Unified Diffusion Reverse Update for Decision-Making

We first recall posterior sampling in contextual bandits. Let $\mathcal{H}_t := \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{t-1}$ denote the interaction history available *before* choosing an action at round t. At round t, given history $\mathcal{H}_t$ and prior $p(\boldsymbol{\theta})$, TS samples a parameter $\widetilde{\boldsymbol{\theta}}_t \sim p(\boldsymbol{\theta} \mid \mathcal{H}_t) \propto p(\boldsymbol{\theta})\, p(\mathcal{H}_t \mid \boldsymbol{\theta})$ and selects the arm $\boldsymbol{x}_t = \arg\max_{\boldsymbol{x}\in\mathcal{X}_t} f(\widetilde{\boldsymbol{\theta}}_t; \boldsymbol{x})$. However, such simple priors cannot capture rich cross-task structure in multi-task settings, leading to misspecified posteriors and suboptimal exploration. To address this, we train a diffusion model on data from past tasks to learn a flexible prior distribution over task parameters. For a new task, at each round we initialize from noise and run an L-step conditional reverse-diffusion process conditioned on the history to produce a posterior sample for arm selection.

We unify diffusion-based posterior sampling for decision-making with a single per-level reverse update (omitting coefficients):

$$\underbrace{\boldsymbol{\theta}_{\ell-1}}_{\text{next state}} \leftarrow \underbrace{\boldsymbol{\theta}_\ell}_{\text{current state}} + \underbrace{\text{unconditional term}}_{\text{diffusion prior drift}} + \underbrace{\text{likelihood term}}_{\text{data drift}} + \underbrace{\text{noise term}}_{\text{randomized exploration}}, \tag{3.1}$$

The unconditional term is the standard reverse step from a pretrained diffusion model, implemented via either a score network or a denoiser network. The likelihood term incorporates bandit interaction history up to round $t-1$ by defining a loss L_t, evaluating it at a diffusion-informed argument $\Phi_\ell(\boldsymbol{\theta}_\ell)$ (the rescaled state or Tweedie estimate), and applying a gradient step $-\eta_\ell \nabla_\theta L_t(\Phi_\ell(\boldsymbol{\theta}_\ell))$. The noise term enables randomized exploration. This decomposition equation 3.1 separates offline knowledge (the diffusion prior drift) from online adaptation (the data drift). From the perspective of fast adaptation, the unconditional diffusion drift transfers the cross-task structure learned from historical tasks, while the likelihood drift uses the small amount of new-task interaction history to specialize the parameter sample to the current task. The noise term preserves the stochasticity required by Thompson Sampling, so that the learner can continue to explore while adapting quickly. For additional motivation behind the use of diffusion models and a mechanism-level illustration of diffusion posterior sampling, we refer readers to Appendix B.1.

In the next two subsections, we instantiate equation 3.1 in two ways. First, **DLTS** follows the TS workflow and performs Langevin updates at each level using the rescaled state, yielding a diffusion-prior approximate sampler for nonlinear reward models. Second, **DPSG** preserves the reverse-diffusion drift and adds a single likelihood drift per level via the Tweedie estimate, resulting in a simple and efficient conditional sampler. Both methods follow the unified diffusion posterior sampling framework in equation 3.1, while each admits its own interpretation and derivation.

3.2 Diffusion Langevin Thompson Sampling

To handle contextual bandits with complex priors, Kveton et al. (2024) proposed DiffTS for linear and generalized linear models with specialized closed-form updates. For nonlinear bandits, such closed forms are unavailable, making algorithm design harder. Motivated by LMC-TS (Xu et al., 2022), an approximate sampling method for nonlinear bandits, we propose **Diffusion Langevin Thompson Sampling (DLTS)** (Algorithm 1). DLTS extends approximate TS to nonlinear rewards (including neural networks) by using a learned diffusion prior.

Algorithm 1 Diffusion Langevin Thompson Sampling (DLTS)

Input: Diffusion denoiser $\varepsilon_{\boldsymbol{\phi}^*}(\cdot,\cdot)$, noise schedule $\{\beta_\ell\}_{\ell=1}^{L+1}$, learning rate $\{\eta_\ell\}_{\ell=1}^{L+1}$.

$\alpha_\ell \leftarrow 1-\beta_\ell$ and $\bar{\alpha}_\ell \leftarrow \prod_{j=1}^{\ell} \alpha_j$ for all ℓ
for $t = 1, 2, \cdots, T$ **do**
  Receive contextual vector $\{\boldsymbol{x}_t(a)\}_{a\in\mathcal{A}}$.
  $(\boldsymbol{\mu}_{L+1}, \boldsymbol{\Sigma}_{L+1}) \leftarrow (\mathbf{0}, \mathbf{I})$.
  for $\ell = L+1, L, \cdots, 1$ **do**
    $\boldsymbol{\theta}_{\ell,0} \leftarrow \boldsymbol{\mu}_\ell$.
    for $k = 0, 1, \cdots, K_\ell - 1$ **do**
      Sample $\boldsymbol{\xi}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.
      $\boldsymbol{\theta}_{\ell,k+1} \leftarrow \big(1 - \frac{(1-\bar{\alpha}_\ell)\eta_\ell}{(1-\bar{\alpha}_{\ell-1})\beta_\ell}\big)\boldsymbol{\theta}_{\ell,k} + \frac{(1-\bar{\alpha}_\ell)\eta_\ell}{(1-\bar{\alpha}_{\ell-1})\beta_\ell}\boldsymbol{\mu}_\ell - \eta_\ell \nabla_{\boldsymbol{\theta}} L_t\big(\boldsymbol{\theta}_{\ell,k}/\sqrt{\bar{\alpha}_{\ell-1}}\big) + \sqrt{2\eta_\ell \zeta_\ell^{-1}}\boldsymbol{\xi}_k$.
    end for
    $\boldsymbol{\theta}_{\ell-1} \leftarrow \boldsymbol{\theta}_{\ell,K_\ell}$.
    $\boldsymbol{\mu}_{\ell-1} \leftarrow \frac{1}{\sqrt{\alpha_{\ell-1}}}\big(\boldsymbol{\theta}_{\ell-1} - \frac{1-\alpha_{\ell-1}}{\sqrt{1-\bar{\alpha}_{\ell-1}}}\varepsilon_{\boldsymbol{\phi}^*}(\boldsymbol{\theta}_{\ell-1}, \ell-1)\big)$.
  end for
  $\widetilde{\boldsymbol{\theta}}_t \leftarrow \boldsymbol{\theta}_0$, take action $\boldsymbol{x}_t = \arg\max_{\boldsymbol{x}_t(a)} f\big(\widetilde{\boldsymbol{\theta}}_t; \boldsymbol{x}_t(a)\big)$.
  Receive reward y_t and update history $\mathcal{H}_{t+1} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{t}$.
end for

Our method follows the TS workflow, which builds a posterior over parameters and samples for arm selection, but it differs in three ways: (1) It replaces a hand-crafted prior with a diffusion prior trained on past tasks. (2) Instead of a single posterior update, it runs a reverse diffusion process and injects interaction history at each level, building the conditional posterior gradually. Formally, the reverse process is a Markov chain with the per-level transition in the form:

$$p(\boldsymbol{\theta}_{\ell-1}|\boldsymbol{\theta}_\ell, \mathcal{H}_t) \propto p(\mathcal{H}_t|\boldsymbol{\theta}_{\ell-1}) p(\boldsymbol{\theta}_{\ell-1}|\boldsymbol{\theta}_\ell), \tag{3.2}$$

where $\ell = L, \ldots, 1$, $\boldsymbol{\theta}_\ell \in \mathbb{R}^d$ is the noisy parameter at level ℓ and the terminal $\boldsymbol{\theta}_0$ is used as the parameter sample for action selection. (3) It applies iterative LMC updates to approximately sample from equation 3.2.

After training the diffusion denoiser $\varepsilon_{\phi^*}(\cdot,\cdot)$ (see Appendix F for more details), we obtain the diffusion model parameter $(\boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)_{\ell\in[L+1]}$ for unconditional sampling in each reverse step: $p(\boldsymbol{\theta}_{\ell-1}|\boldsymbol{\theta}_\ell) = \mathcal{N}(\boldsymbol{\theta}_{\ell-1}; \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)$, where $\boldsymbol{\mu}_\ell = \frac{1}{\sqrt{\alpha_\ell}}(\boldsymbol{\theta}_\ell - \frac{1-\alpha_\ell}{\sqrt{1-\bar{\alpha}_\ell}}\varepsilon_{\phi^*}(\boldsymbol{\theta}_\ell, \ell))$, $\boldsymbol{\Sigma}_\ell = \frac{1-\bar{\alpha}_{\ell-1}}{1-\bar{\alpha}_\ell}\beta_\ell \mathbf{I}$. Note that at step t, the nonlinear reward model is defined as $y_t = f(\boldsymbol{\theta}^*; \boldsymbol{x}_t) + \varepsilon_t$. When we assume Gaussian noise $\varepsilon_t \sim \mathcal{N}(0, \sigma^2)$, then the likelihood becomes $p(\mathcal{H}_t|\boldsymbol{\theta}) \propto \exp(-\sigma^{-2}\sum_{i=1}^t (f(\boldsymbol{\theta}; \boldsymbol{x}_i) - y_i)^2) = \exp(-L_t(\boldsymbol{\theta}))$, where $L_t(\boldsymbol{\theta}) = \sigma^{-2}\sum_{i=1}^t (f(\boldsymbol{\theta}; \boldsymbol{x}_i) - y_i)^2$. Therefore, we can approximate the posterior in each reverse step for conditional sampling as follows (refer to Appendix B.2 for derivation)

$$p(\boldsymbol{\theta}_{\ell-1}|\boldsymbol{\theta}_\ell, \mathcal{H}_t) \propto \exp(-L_t(\boldsymbol{\theta}_{\ell-1}/\sqrt{\bar{\alpha}_{\ell-1}})) \cdot \mathcal{N}(\boldsymbol{\theta}_{\ell-1}; \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell).$$

To sample from $p(\boldsymbol{\theta}_{\ell-1}|\boldsymbol{\theta}_\ell, \mathcal{H}_t)$, we apply Langevin Monte Carlo for approximate sampling. Specifically, at round t and reverse step ℓ, we iteratively conduct Langevin updates

$$\boldsymbol{\theta}_{\ell,k+1} = \boldsymbol{\theta}_{\ell,k} - \eta_\ell\Big[\nabla_{\boldsymbol{\theta}} L_t\big(\boldsymbol{\theta}_{\ell,k}/\sqrt{\bar{\alpha}_{\ell-1}}\big) + \boldsymbol{\Sigma}_\ell^{-1}\big(\boldsymbol{\theta}_{\ell,k} - \boldsymbol{\mu}_\ell\big)\Big] + \sqrt{2\eta_\ell\zeta_\ell^{-1}}\boldsymbol{\xi}_k,$$

where $\boldsymbol{\xi}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, η_ℓ is the step size, and ζ_ℓ is the temperature. We initialize $\boldsymbol{\theta}_{\ell,0} = \boldsymbol{\mu}_\ell$, after K_ℓ iterations, we set $\boldsymbol{\theta}_{\ell-1} = \boldsymbol{\theta}_{\ell,K_\ell}$ as the sample from the reverse step. Specifically, when we only conduct one LMC update ($K_\ell = 1$), we have the reverse update

$$\boldsymbol{\theta}_{\ell-1} \leftarrow \left(1 - \tfrac{(1-\bar{\alpha}_\ell)\eta_\ell}{(1-\bar{\alpha}_{\ell-1})\beta_\ell}\right)\boldsymbol{\theta}_\ell + \tfrac{(1-\bar{\alpha}_\ell)\eta_\ell}{(1-\bar{\alpha}_{\ell-1})\beta_\ell}\boldsymbol{\mu}_\ell - \eta_\ell\nabla_{\boldsymbol{\theta}} L_t\big(\boldsymbol{\theta}_\ell/\sqrt{\bar{\alpha}_{\ell-1}}\big) + \sqrt{2\eta_\ell\zeta_\ell^{-1}}\boldsymbol{\xi}. \tag{3.3}$$

Note that equation 3.3 aligns with our unified update equation 3.1. This confirms the rationality of our framework and the essence of diffusion-based posterior sampling. Intuitively, DLTS explicitly forms a conditional posterior by incorporating the history into the diffusion prior at each reverse step, then draws samples from it via approximate sampling.

3.3 Diffusion Posterior Sampling with Guidance

We now give a simpler and faster realization inspired by Diffusion Posterior Sampling (DPS) (Chung et al., 2023), a common approach for conditional sampling in inverse problems. DPS augments the unconditional reverse step with a likelihood score term and then runs the reverse chain, yielding a conditional sampler. With a pretrained unconditional process, drawing a parameter sample for arm selection is straightforward: run the unconditional reverse steps and add a likelihood drift for guidance. This requires only the unconditional score and a tractable proxy for the likelihood at the Tweedie estimate. Following this, we propose **Diffusion Posterior Sampling with Guidance (DPSG)** in Algorithm 2.

Algorithm 2 Diffusion Posterior Sampling with Guidance (DPSG)

Input: score network $s_{\psi^*}(\cdot,\cdot)$, noise schedule $\{\beta_\ell\}_{\ell=1}^L$, learning rate $\{\eta_\ell\}_{\ell=1}^L$.

1: $\alpha_\ell \leftarrow 1 - \beta_\ell$ and $\bar{\alpha}_\ell \leftarrow \prod_{j=1}^\ell \alpha_j$ for all ℓ.
2: **for** $t = 1, 2, \cdots, T$ **do**
3: Receive contextual vector $\{\boldsymbol{x}_t(a)\}_{a\in\mathcal{A}}$.
4: $\boldsymbol{\theta}_L \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.
5: **for** $\ell = L, L-1, \cdots, 1$ **do**
6: $\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_\ell, \ell) \leftarrow \frac{1}{\sqrt{\bar{\alpha}_\ell}}\big(\boldsymbol{\theta}_\ell + (1-\bar{\alpha}_\ell)s_{\psi^*}(\boldsymbol{\theta}_\ell, \ell)\big)$.
7: $\boldsymbol{z}_\ell \sim \mathcal{N}(\mathbf{0}, \beta_\ell\mathbf{I})$.
8: $\boldsymbol{\theta}_{\ell-1} \leftarrow \frac{1}{\sqrt{\alpha_\ell}}\boldsymbol{\theta}_\ell + \frac{\beta_\ell}{\sqrt{\bar{\alpha}_\ell}}s_{\psi^*}(\boldsymbol{\theta}_\ell, \ell) - \eta_\ell\nabla L_t\big(\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_\ell, \ell)\big) + \boldsymbol{z}_\ell$.
9: **end for**
10: $\widetilde{\boldsymbol{\theta}}_t \leftarrow \boldsymbol{\theta}_0$, take action $\boldsymbol{x}_t = \arg\max_{\boldsymbol{x}_t(a)} f\big(\widetilde{\boldsymbol{\theta}}_t; \boldsymbol{x}_t(a)\big)$.
11: Receive reward y_t and update history $\mathcal{H}_{t+1} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^t$.
12: **end for**

To derive diffusion reverse update (Line 8 in Algorithm 2), we follow Chung et al. (2023) and decompose the conditional score into two terms,

$$\nabla_{\boldsymbol{\theta}} \log p_s(\boldsymbol{\theta}_s|\mathcal{H}_t) = \underbrace{\nabla_{\boldsymbol{\theta}} \log p_s(\boldsymbol{\theta}_s)}_{\text{unconditional score}} + \underbrace{\nabla_{\boldsymbol{\theta}} \log p_s(\mathcal{H}_t|\boldsymbol{\theta}_s)}_{\text{likelihood score}}.$$

For the first term, we can use pretrained diffusion model score network s_{ψ^*} to approximate the unconditional score. For the second term, we approximate the likelihood score via Tweedie's formula. Since $p(\mathcal{H}_t|\boldsymbol{\theta}) \propto \exp(-L_t(\boldsymbol{\theta}))$, based on Tweedie's formula (Efron, 2011), we can obtain a tractable approximation for $\nabla_{\boldsymbol{\theta}} \log p_s(\mathcal{H}_t|\boldsymbol{\theta}_s)$ (refer to Appendix B.3 for details of derivation): $\nabla_{\boldsymbol{\theta}} \log p_s(\mathcal{H}_t|\boldsymbol{\theta}_s) \simeq -\nabla_{\boldsymbol{\theta}} L_t(\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_s, s))$, where $\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_s, s) \simeq 1/\sqrt{\bar{\alpha}_s}(\boldsymbol{\theta}_s + (1-\bar{\alpha}_s)s_{\psi^*}(\boldsymbol{\theta}_s, s))$. Therefore, we can further have the approximation of the conditional score of reverse dynamics,

$$\nabla_{\boldsymbol{\theta}} \log p_s(\boldsymbol{\theta}_s|\mathcal{H}_t) \simeq s_{\psi^*}(\boldsymbol{\theta}_s, s) - \nabla_{\boldsymbol{\theta}} L_t\big(\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_s, s)\big). \tag{3.4}$$

To obtain a discrete-time algorithm, we use DDPM ancestral sampling (Ho et al., 2020) to implement conditional sampling via the conditional score equation 3.4. The reverse update is done by two steps. First, we use the unconditional score $s_{\psi^*}(\boldsymbol{\theta}_\ell, \ell)$ through Tweedie's estimate $\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_\ell, \ell)$ to form the Gaussian posterior mean, then we sample

$$\boldsymbol{\theta}'_{\ell-1} \leftarrow \tfrac{\sqrt{\alpha_\ell}(1-\bar{\alpha}_{\ell-1})}{1-\bar{\alpha}_\ell}\boldsymbol{\theta}_\ell + \tfrac{\sqrt{\bar{\alpha}_{\ell-1}}\beta_\ell}{1-\bar{\alpha}_\ell}\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_\ell, \ell) + \boldsymbol{z}_\ell, \tag{3.5}$$

where $\boldsymbol{z}_\ell \sim \mathcal{N}(\mathbf{0}, \beta_\ell \mathbf{I})$. Based on equation 3.4, the second step is that we make this update conditional by subsequently adding one likelihood-based term. In practice, after sampling $\boldsymbol{\theta}'_{\ell-1}$, we take a gradient step using the likelihood L_t at the Tweedie estimate for guidance: $\boldsymbol{\theta}_{\ell-1} \leftarrow \boldsymbol{\theta}'_{\ell-1} - \eta_\ell \nabla_{\boldsymbol{\theta}} L_t(\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_\ell, \ell))$. When we merge this with equation 3.5, we obtain line 8 in Algorithm 2,

$$\boldsymbol{\theta}_{\ell-1} \leftarrow \tfrac{1}{\sqrt{\alpha_\ell}}\boldsymbol{\theta}_\ell + \tfrac{\beta_\ell}{\sqrt{\alpha_\ell}}s_{\psi^*}(\boldsymbol{\theta}_\ell, \ell) - \eta_\ell \nabla_{\boldsymbol{\theta}} L_t\big(\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_\ell, \ell)\big) + \boldsymbol{z}_\ell. \tag{3.6}$$

Note that equation 3.6 aligns with our unified update equation 3.1. This confirms the rationality of our framework and the essence of diffusion-based posterior sampling. However, from the intuition perspective, DPSG is to use history information as guidance after unconditional update based on the diffusion prior. This is different from DLTS, which directly integrates history information into the unconditional update based on the diffusion prior to achieve conditional sampling.

Empirical Variant of DPSG. DPSG is simple and fast, but it relies on the accuracy of likelihood-score at the Tweedie estimate. A recent work Xu et al. (2025) observes that DPS has the properties of high bias and low diversity, thus behaving like an implicit, but unstable, MAP estimator. Therefore, multi-step projection is used to solve this MAP optimization problem. Motivated by Xu et al. (2025), to stabilize learning, we introduce **DPSG with Multi-step Projection (DPSG-MP)** (Algorithm 3 in Appendix B.4), which is a practical DPSG variant that replaces the single correction with a short inner loop of gradient ascent on the likelihood score, and after each step projects the iterate onto the sphere where the reverse transition $p_{\psi^*}(\boldsymbol{\theta}_{\ell-1}|\boldsymbol{\theta}_\ell)$ concentrates around. Specifically, we first initialize $\boldsymbol{\theta}_{\ell-1,0} \leftarrow 1/\sqrt{\alpha_\ell}\boldsymbol{\theta}_\ell + \beta_\ell/\sqrt{\alpha_\ell}s_{\psi^*}(\boldsymbol{\theta}_\ell, \ell) + \boldsymbol{z}_\ell$, then we iteratively update

$$\boldsymbol{\theta}_{\ell-1,k+1} \leftarrow \mathrm{Proj}(\boldsymbol{\theta}_{\ell-1,k} - \eta_\ell \nabla_{\boldsymbol{\theta}} L_t(\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_\ell, \ell))),$$

where $k = 0, 1, \ldots, K_\ell$, function $\mathrm{Proj}(\cdot)$ projects the input onto sphere surface $\mathcal{S}(p_{\psi^*}(\boldsymbol{\theta}_{\ell-1}|\boldsymbol{\theta}_\ell))$, with radius $r_\ell = \|\boldsymbol{z}_\ell\|_2$ and center at $\mathbb{E}[\boldsymbol{\theta}_{\ell-1}|\boldsymbol{\theta}_\ell]$. Finally, we assign $\boldsymbol{\theta}_{\ell-1,K_\ell}$ to $\boldsymbol{\theta}_{\ell-1}$ and finish the update. We provide the complete algorithm and further explanation in Appendix B.4.

4 Theoretical Justification

We analyze the connection between Thompson sampling and diffusion Thompson Sampling.

Definition 4.1. We define the following oracle algorithm as **Oracle Thompson Sampling (OTS)**, which applies exact posterior sampling and greedy policy according to the true prior and likelihood:

1) Use true prior p_0.
2) Maintain exact posterior in each round: $p_t(\boldsymbol{\theta}) \equiv p(\boldsymbol{\theta}|\mathcal{H}_t) \propto p_0(\boldsymbol{\theta}) \prod_{i=1}^{t-1} p(y_i|\boldsymbol{\theta}, \boldsymbol{x}_i)$.
3) Sample $\boldsymbol{\theta}_t^{\mathrm{OTS}} \sim p_t(\boldsymbol{\theta})$, then select arm $\boldsymbol{x}_t = \arg\max_{\boldsymbol{x} \in \mathcal{X}_t} f(\boldsymbol{\theta}_t^{\mathrm{OTS}}; \boldsymbol{x})$.

Definition 4.2. We define the **Oracle Diffusion Thompson Sampling (ODTS)** algorithm, which performs posterior sampling via a reverse-diffusion chain using the same pretrained diffusion model as DLTS (Algorithm 1).

1) Train a diffusion reverse process to learn prior p_0 and obtain score/denoiser network $s_{\psi^*}(\cdot, \cdot)$.
2) The per-level conditional reverse process is implemented exactly: $\widetilde{p}_t(\boldsymbol{\theta}_{0:L}) \equiv p(\boldsymbol{\theta}_{0:L}|\mathcal{H}_t) = p(\boldsymbol{\theta}_L|\mathcal{H}_t) \prod_{\ell=1}^{L} p(\boldsymbol{\theta}_{\ell-1}|\boldsymbol{\theta}_\ell, \mathcal{H}_t)$ and $\widetilde{p}_t(\boldsymbol{\theta}) = \int \widetilde{p}_t(\boldsymbol{\theta}_{0:L}) d\boldsymbol{\theta}_{1:L} = \int p(\boldsymbol{\theta}_{0:L}|\mathcal{H}_t) d\boldsymbol{\theta}_{1:L}$.

3) Sample $\boldsymbol{\theta}_t^{\text{ODTS}} \sim \widetilde{p}_t(\boldsymbol{\theta})$, then selects arm $\boldsymbol{x}_t = \arg\max_{\boldsymbol{x} \in \mathcal{X}_t} f(\boldsymbol{\theta}_t^{\text{ODTS}}; \boldsymbol{x})$.

We make the following assumption to measure the accuracy of score estimation.

Assumption 4.3. For any diffusion time $s \in [0, \mathcal{S}]$ and let p_s denote the s-marginal distribution of the forward diffusion over parameters $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^d$. Let $s_{\boldsymbol{\psi}^*}(\boldsymbol{\theta}, s)$ be the learned score estimator. Assume there exists ϵ_{score} such that for all s and $\boldsymbol{\theta}$, $\|s_{\boldsymbol{\psi}^*}(\boldsymbol{\theta}, s) - \nabla_{\boldsymbol{\theta}} \log p_s(\boldsymbol{\theta})\|_2 \leq \epsilon_{\text{score}}$.

Note that the key distinction between OTS and ODTS is whether the reverse-diffusion process is trained to recover the true prior p_0. The theorem below shows that if ODTS learns the prior exactly (i.e $\epsilon_{\text{score}} = 0$ in Assumption 4.3), then OTS and ODTS are equivalent.

Theorem 4.4. If ODTS perfectly trains a diffusion reverse process to recover the true prior p_0 and the score/denoiser network matches the true score at every diffusion level ℓ, i.e., $s_{\boldsymbol{\psi}^*}(\cdot, \ell) \equiv \nabla_{\boldsymbol{\theta}} \log p_\ell(\cdot)$, then the oracle algorithms OTS and ODTS are equivalent in the sense that they induce the same posterior distribution $\widetilde{p}_t(\boldsymbol{\theta}) = p_t(\boldsymbol{\theta})$ and the same arm-selection distribution.

Remark 4.5. Theorem 4.4 confirms that our diffusion-based framework is mathematically consistent with standard Thompson Sampling in the ideal limit, implying that any performance gaps observed in practice are due to approximation errors (e.g., score estimation or discretization).

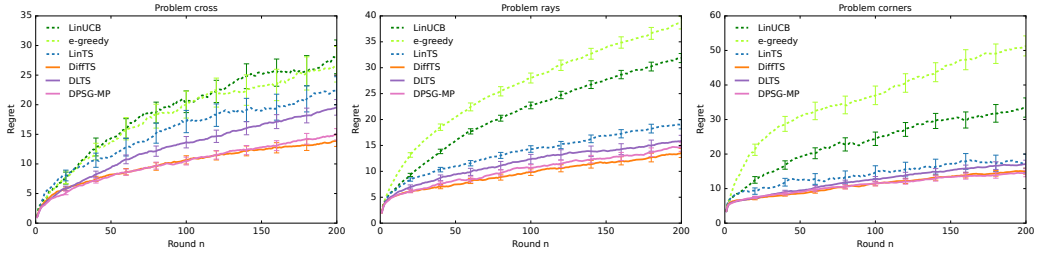
5 Experiments

In this section, we conduct experiments to evaluate DLTS and a variant of DPSG named DPSG-MP (refer to Appendix B.4). We aim to demonstrate their effectiveness in both linear and nonlinear contextual bandit settings. We additionally provide the real-world application results on MovieLens in Appendix H.3 due to the space limit.

5.1 Simulation

Our diffusion prior is implemented within a Denoising Diffusion Probabilistic Model (DDPM) framework, using the standard ϵ-prediction parameterization. The denoiser network is an MLP that incorporates time embeddings. For synthetic simulations, we use $L = 100$ diffusion steps and a linear noise schedule for β_ℓ. More details can be found in Appendix G.

Prior Distribution. We follow Kveton et al. (2024) to design the prior over task parameters. We consider six prior distributions, including 'cross', 'rays', 'triangles', 'swirl', 'H' and 'corners'. Each prior distribution has 10000 samples with $d = 2$ parameters defined as $\boldsymbol{\theta}$. We use 80% of the samples as the training samples, which refer to the previous task. For the remaining 20% samples, we use them as the test samples, which refer to the new tasks. We illustrate the prior distribution in Figure 5 in Appendix G.2. Based on the prior distribution, we design linear and non-linear contextual bandits with different reward models.

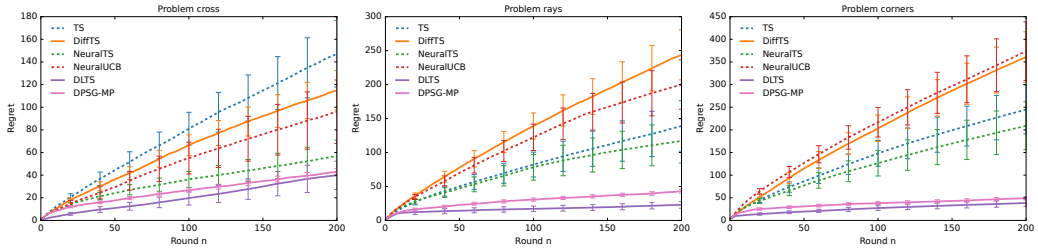


(a) cross (b) rays (c) corners

Figure 1: Performance of all algorithms on the linear contextual bandit tasks for the three prior distributions. Our proposed methods achieve cumulative regret comparable to LinTS and DiffTS. More results can be found in Appendix H.

Linear Bandit. We first evaluate our methods in the linear contextual bandit setting. For each task, the ground-truth parameter $\boldsymbol{\theta}$ is drawn from the test set of a given prior distribution, and rewards are generated according to the model $r = \boldsymbol{x}^\top \boldsymbol{\theta} + \varepsilon$ where $\varepsilon \sim \mathcal{N}(0, 1)$. We compare our proposed DLTS and DPSG-MP which is a more stable version of DPSG with several baseline methods, including the

LinTS (Agrawal & Goyal, 2013), LinUCB (Chu et al., 2011), ϵ-greedy and DiffTS (Kveton et al., 2024). For LinTS, LinUCB, and ϵ-greedy algorithms, we directly conduct them in the test bandit task. Our methods and DiffTS leverage a diffusion prior trained on the 80% training partition of the parameter data. Performance is measured over 64 new tasks per trial, with each task running for a horizon of 200 steps. To ensure statistical robustness, all results are averaged over 8 independent trials. From the regret comparison in Figure 1, we observe that our methods DLTS and DPSG-MP achieve cumulative regret that is highly competitive with the specialized baseline algorithms. This is a noteworthy outcome, as methods like LinTS and DiffTS are designed specifically for the linear setting and leverage efficient closed-form posterior updates. In contrast, our framework relies on a more general, gradient-based sampling approach. Subsequently, we show that our methods can perform well in non-linear settings where these specialized closed-form solutions can not address. The good performance of the algorithms demonstrates the benefits of leveraging the diffusion prior learned on the previous task in solving the new tasks with the same prior distribution.



(a) cross (b) rays (c) corners

Figure 2: Performance of all algorithms on the sigmoid-gated nonlinear bandit tasks for the three prior distributions. Our proposed methods outperform strong baselines achieving the lowest cumulative regret. More results can be found in Appendix H.

Nonlinear Bandit. We also evaluate our framework in scenarios where the reward function is inherently nonlinear. We construct three such environments where the reward is defined by $r = f(\boldsymbol{x}^\top \boldsymbol{\theta}) + \varepsilon$, with $\varepsilon \sim \mathcal{N}(0, 1)$. The nonlinear functions $f(\cdot)$ we consider are a cosine model ($f(z) = \cos(3z)$), a quadratic model ($f(z) = z^2$), and a sigmoid-gated model ($f(z) = 2z \cdot \text{sigmoid}(z) + 1$). Due to space limits, results for the cosine model and the quadratic model are deferred to Appendix H.2.

In the nonlinear setting, our methods DLTS and DPSG-MP, use an MLP to represent the reward function. A key distinction from prior sections is that the diffusion model is now trained to learn a prior directly over the parameters of this MLP, using network weights from past tasks as training data. This setup tests the ability of our framework to handle priors in the high-dimensional weight space of neural networks. We compare our proposed algorithms with several baseline methods, including the LinTS, DiffTS, NeuralTS (Zhang et al., 2021) and NeuralUCB (Zhou et al., 2020). Visualizations of the diffusion priors over MLP parameters are provided in Figure 8 in Appendix H.1.

When tested on the more complex sigmoid-gated reward model, our methods demonstrate a clear advantage, as illustrated in Figure 2. Both DLTS and DPSG-MP consistently outperform the strong NeuralTS and NeuralUCB baselines, achieving the lowest cumulative regret. This superior performance underscores the primary benefit of our framework. By leveraging a powerful, pretrained diffusion prior, our algorithms can explore the parameter space more effectively and adapt more quickly in challenging nonlinear environments. This stands in contrast to the baseline methods, which must learn from scratch on each new task. The plotted results show the mean cumulative regret and standard error averaged over 64 tasks across 8 independent trials.

5.2 Wall-Clock Time Analysis and Ablation Study

Wall-Clock Time Analysis. To rigorously assess the trade-off between computational cost and sample efficiency, we perform a time-to-accuracy analysis. This metric effectively normalizes for sample efficiency, demonstrating that a computationally more intensive algorithm (such as ours) can achieve superior overall time efficiency if it requires significantly fewer interactions to reach a high-quality policy compared to faster but less data-efficient baselines. We benchmark all algorithms in the nonlinear sigmoid-gated bandit setting using the ‘triangles’ prior distribution. We conduct our

DLTS and DPSG-MP with inner-loop K_ℓ to be 10 and 1 respectively. We define a target performance threshold of 0.5 average regret (calculated as cumulative regret divided by the number of rounds refer to Figure 3). For each algorithm, we measure the total wall-clock time required to achieve this threshold. A maximum time budget of 3000 seconds is enforced. For fair comparison, all evaluations are conducted on an Intel Xeon E5-2640 v4 CPU. The results, summarized in Table 1 and Figure 3, show that our methods (DLTS and DPSG-MP) reach the performance threshold significantly faster than the baselines due to their superior sample efficiency.

Comparison with Learned-Prior Baselines. We examine whether diffusion priors provide an advantage over learned Gaussian priors from the same historical tasks. We compare with MixTS (Hong et al., 2022), which learns a two-component Gaussian mixture prior over the underlying task parameters, and Learned-Gaussian TS, which learns a Gaussian prior over the historical MLP parameters and uses LMC for posterior sampling. DPSG-MP achieves lower mean regret than both baselines on all six problems. Details can be found in Appendix H.4.

Table 1: Time to reach an average regret of 0.5 on Triangles over 64 tasks. Our methods reach the threshold substantially faster than the baselines.

Algorithm	Round	Time (s)
DiffTS	> 2000	> 3000
NeuralTS	1505	> 3000
NeuralUCB	799	~ 1967.71
DLTS	**93**	$\sim$ **257.67**
DPSG-MP	**54**	$\sim$ **173.63**

High-Dimensional Neural Reward Models. We examine whether our results depend on the low-dimensional structure induced by the two-dimensional task parameters. We directly sample 1185-dimensional MLP parameters from a four-component full-rank Gaussian mixture and compare with NeuralTS and NeuralUCB. DLTS and DPSG-MP achieve lower mean regret than both baselines at $T = 50$ and $T = 200$. Details can be found in Appendix H.5.

Sensitivity to Diffusion Steps and Inner-Loop Updates. We defer detailed ablations on diffusion steps L and inner-loop updates K_ℓ to Appendix H.6 and Appendix H.7. Overall, larger L improves posterior sampling quality, while a small number of inner-loop updates is generally sufficient.

Long-Horizon Results. We defer the long-horizon evaluation under the sigmoid-gated nonlinear bandit setting to Appendix H.8. Overall, our methods remain competitive over $T = 2000$ rounds across all six priors, suggesting that diffusion posterior sampling provides stable exploration benefits beyond early-stage adaptation.

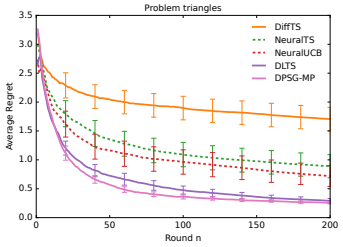


Figure 3: Average regret on Triangles. DLTS and DPSG-MP achieve the lowest regret.

Sensitivity to Amount of Historical Data. We study how the amount of historical data used to train the diffusion prior affects downstream adaptation. Our main experiments use 8000 historical task parameters for prior learning. We additionally train separate diffusion priors using 10%, 50%, and 100% of these training data and evaluate DLTS and DPSG-MP on the sigmoid-gated Swirl and Triangles environments. As shown in Table 11 in Appendix H.9, using only 10% of the historical data generally leads to higher regret, while using 50% often approaches the performance obtained with the full training set.

6 Conclusion and Limitations

We studied fast adaptation in multi-task nonlinear contextual bandits by learning a flexible diffusion prior from past tasks and using it for posterior sampling on a new task with limited online interactions. We developed a unified diffusion reverse update framework for decision-making, instantiated as **DLTS** and **DPSG**, and theoretically analyzed its connection to standard Thompson Sampling. Across linear and non-linear contextual bandits, our methods are competitive with closed-form baselines in the linear regime and achieve consistent gains in neural settings. Our wall-clock time analysis further shows that the learned diffusion prior can reduce both the number of online interactions and the total time required to reach a target performance level, highlighting the potential of diffusion posterior sampling for fast adaptation in nonlinear online decision-making. Although the L-step diffusion reverse process increases per-step latency, this cost can be fully offset in practice by the substantially fewer online interactions needed for fast adaptation.

AI USE STATEMENT

In this work, we used generative AI tools only for coding and polishing the writing. We have not used generative AI tools for research ideation, methodology or experiment design, theoretical development or proofs, data generation or analysis, or other tasks with required disclosure. We have reviewed all AI-assisted work. AI-assisted code was verified and tested for correctness, and AI-polished writing was reviewed and revised by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we provide the source code and implementation in the supplementary materials. The main paper describes the proposed methods, training procedures, experimental setups, evaluation protocols, and the datasets and environments used in our experiments. Additional implementation details, hyperparameters, and experimental configurations are provided in the appendix. For the theoretical results, we state the required assumptions and provide complete proofs in the appendix.

REFERENCES

Shipra Agrawal and Navin Goyal. Thompson sampling for contextual bandits with linear payoffs. In Sanjoy Dasgupta and David McAllester (eds.), *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pp. 127–135, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR. URL https://proceedings.mlr.press/v28/agrawal13.html.

Shipra Agrawal and Navin Goyal. Near-optimal regret bounds for thompson sampling. *Journal of the ACM (JACM)*, 64(5):1–24, 2017.

Anurag Ajay, Yilun Du, Abhi Gupta, Joshua B. Tenenbaum, Tommi S. Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision making? In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=sP1fo2K9DFG.

Imad Aouali. Diffusion models meet contextual bandits with large action spaces. *arXiv preprint arXiv:2402.10028*, 2024.

Yikun Ban, Yuchen Yan, Arindam Banerjee, and Jingrui He. Ee-net: Exploitation-exploration neural networks in contextual bandits. *arXiv preprint arXiv:2110.03177*, 2021.

Sam Bond-Taylor, Adam Leach, Yang Long, and Chris G Willcocks. Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models. *IEEE transactions on pattern analysis and machine intelligence*, 44(11):7327–7347, 2021.

Sébastien Bubeck, Nicolo Cesa-Bianchi, and Gábor Lugosi. Bandits with heavy tail. *IEEE Transactions on Information Theory*, 59(11):7711–7717, 2013.

Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. *Advances in neural information processing systems*, 24, 2011.

Wei Chu, Lihong Li, Lev Reyzin, and Robert Schapire. Contextual bandits with linear payoff functions. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 208–214. JMLR Workshop and Conference Proceedings, 2011.

Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=OnD9zGAGT0k.

Pierre Clavier, Tom Huix, and Alain Oliviero Durmus. VITS : Variational inference thompson sampling for contextual bandits. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st*

International Conference on Machine Learning, volume 235 of *Proceedings of Machine Learning Research*, pp. 9033–9075. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/clavier24a.html.

Zehao Dou and Yang Song. Diffusion posterior sampling for linear inverse problem solving: A filtering perspective. In *The Twelfth International Conference on Learning Representations*, 2024.

Bradley Efron. Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106(496):1602–1614, 2011.

Chelsea Finn, Kelvin Xu, and Sergey Levine. Probabilistic model-agnostic meta-learning. *Advances in neural information processing systems*, 31, 2018.

Florence Forbes and Darren Wraith. A new family of multivariate heavy-tailed distributions with variable marginal amounts of tailweight: application to robust clustering. *Statistics and computing*, 24(6):971–984, 2014.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

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

Joey Hong, Branislav Kveton, Manzil Zaheer, Mohammad Ghavamzadeh, and Craig Boutilier. Thompson sampling with a mixture prior. In *International Conference on Artificial Intelligence and Statistics*, pp. 7565–7586. PMLR, 2022.

Yu-Guan Hsieh, Shiva Kasiviswanathan, Branislav Kveton, and Patrick Blöbaum. Thompson sampling with diffusion generative prior. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 13434–13468. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/hsieh23a.html.

Hao-Lun Hsu, Weixin Wang, Miroslav Pajic, and Pan Xu. Randomized exploration in cooperative multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 37: 74617–74689, 2024.

Haque Ishfaq, Qiwen Cui, Viet Nguyen, Alex Ayoub, Zhuoran Yang, Zhaoran Wang, Doina Precup, and Lin Yang. Randomized exploration in reinforcement learning with general value function approximation. In *International Conference on Machine Learning*, pp. 4607–4616. PMLR, 2021.

Haque Ishfaq, Qingfeng Lan, Pan Xu, A. Rupam Mahmood, Doina Precup, Anima Anandkumar, and Kamyar Azizzadenesheli. Provable and practical: Efficient exploration in reinforcement learning via langevin monte carlo. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=nfIAEJFiBZ.

Haque Ishfaq, Yixin Tan, Yu Yang, Qingfeng Lan, Jianfeng Lu, A. Rupam Mahmood, Doina Precup, and Pan Xu. More efficient randomized exploration for reinforcement learning via approximate sampling. *Reinforcement Learning Journal*, 3:1211–1235, 2024b.

Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 9902–9915. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/janner22a.html.

David Janz, Alexander Litvak, and Csaba Szepesvári. Ensemble sampling for linear bandits: small ensembles suffice. *Advances in Neural Information Processing Systems*, 37:23679–23704, 2024.

Tianyuan Jin, Pan Xu, Jieming Shi, Xiaokui Xiao, and Quanquan Gu. Mots: Minimax optimal thompson sampling. In *International Conference on Machine Learning*, pp. 5074–5083. PMLR, 2021.

Tianyuan Jin, Xianglin Yang, Xiaokui Xiao, and Pan Xu. Thompson sampling with less exploration is fast and optimal. In *International Conference on Machine Learning*, pp. 15239–15261. PMLR, 2023.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Branislav Kveton, Manzil Zaheer, Csaba Szepesvari, Lihong Li, Mohammad Ghavamzadeh, and Craig Boutilier. Randomized exploration in generalized linear bandits. In *International Conference on Artificial Intelligence and Statistics*, pp. 2066–2076. PMLR, 2020.

Branislav Kveton, Boris N. Oreshkin, Youngsuk Park, Aniket Anand Deshmukh, and Rui Song. Online posterior sampling with a diffusion prior. In *Advances in Neural Information Processing Systems*, 2024.

Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

Harin Lee and Min-hwan Oh. Improved regret of linear ensemble sampling. *Advances in Neural Information Processing Systems*, 37:92803–92831, 2024.

Ji Li and Chao Wang. Efficient diffusion posterior sampling for noisy inverse problems. *SIAM Journal on Imaging Sciences*, 18(2):1468–1492, 2025.

Haohe Liu, Zehua Chen, Yi Yuan, Xinhao Mei, Xubo Liu, Danilo Mandic, Wenwu Wang, and Mark D Plumbley. AudioLDM: Text-to-audio generation with latent diffusion models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 21450–21474. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/liu23f.html.

Xiuyuan Lu and Benjamin Van Roy. Ensemble sampling. *Advances in neural information processing systems*, 30, 2017.

Eric Mazumdar, Aldo Pacchiano, Yian Ma, Michael Jordan, and Peter Bartlett. On approximate thompson sampling with langevin algorithms. In *international conference on machine learning*, pp. 6797–6807. PMLR, 2020.

Ian Osband, Daniel Russo, and Benjamin Van Roy. (more) efficient reinforcement learning via posterior sampling. *Advances in Neural Information Processing Systems*, 26, 2013.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021.

Chao Qin, Zheng Wen, Xiuyuan Lu, and Benjamin Van Roy. An analysis of ensemble sampling. *Advances in Neural Information Processing Systems*, 35:21602–21614, 2022.

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

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.

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

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=PxTIG12RRHS.

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

Risto Vuorio, Shao-Hua Sun, Hexiang Hu, and Joseph J Lim. Multimodal model-agnostic meta-learning via task-aware modulation. *Advances in neural information processing systems*, 32, 2019.

Zihui Wu, Yu Sun, Yifan Chen, Bingliang Zhang, Yisong Yue, and Katherine Bouman. Principled probabilistic imaging using diffusion models as plug-and-play priors. *Advances in Neural Information Processing Systems*, 37:118389–118427, 2024.

Pan Xu, Hongkai Zheng, Eric V Mazumdar, Kamyar Azizzadenesheli, and Animashree Anandkumar. Langevin monte carlo for contextual bandits. In *International Conference on Machine Learning*, pp. 24830–24850. PMLR, 2022.

Tongda Xu, Xiyan Cai, Xinjie Zhang, Xingtong Ge, Dailan He, Ming Sun, Jingjing Liu, Ya-Qin Zhang, Jian Li, and Yan Wang. Rethinking diffusion posterior sampling: From conditional score estimator to maximizing a posterior. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=GcvLoqOoXL.

Weitong Zhang, Dongruo Zhou, Lihong Li, and Quanquan Gu. Neural thompson sampling. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=tkAtoZkcUnm.

Haoyang Zheng, Wei Deng, Christian Moya, and Guang Lin. Accelerating approximate thompson sampling with underdamped langevin monte carlo. In *International Conference on Artificial Intelligence and Statistics*, pp. 2611–2619. PMLR, 2024.

Dongruo Zhou, Lihong Li, and Quanquan Gu. Neural contextual bandits with UCB-based exploration. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 11492–11502. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/zhou20a.html.

A Related Work

Diffusion Models Generative models have achieved strong results in modeling complex, multi-modal distributions (Kingma & Welling, 2013; Goodfellow et al., 2014; Papamakarios et al., 2021; Ho et al., 2020). Among them, diffusion models are a good fit for inverse problems and decision-making because their reverse-time iterative sampling naturally supports conditioning on observed data (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021). An inverse problem is to seek unknown parameters or signals from indirect, noisy observations using a forward measurement model. In this area, Chung et al. (2023) is a cornerstone: it samples from a diffusion-model posterior by augmenting the score with the observation likelihood. Follow-up work improves stability (Xu et al., 2025), image quality (Dou & Song, 2024), learning efficiency (Li & Wang, 2025), and offers principled formulations with theoretical guarantees that avoid inaccurate approximations (Wu et al., 2024). While most of these focus on imaging, they provide useful insight for sequential decision making. In offline settings, conditional diffusion has been used to synthesize trajectories with strong results on standard benchmarks (Janner et al., 2022; Ajay et al., 2023). In online settings, Hsieh et al. (2023) studied multi-armed bandits, learn a flexible diffusion prior over task parameters, and couple it with Thompson Sampling, including variance calibration for reliable uncertainty. Kveton et al. (2024) adapted diffusion priors to linear and generalized linear contextual bandits via closed-form reverse updates. Aouali (2024) studied contextual bandits with a linear diffusion-model prior but the model reduces to a linear Gaussian case rather than a general diffusion model under this assumption. However, multi-task nonlinear contextual bandits remain underexplored. The core challenge is the absence of closed-form reverse updates for nonlinear reward models, which complicates algorithm design. Without closed forms, sampling error can be large; coupled with limited interactions per task and possible inaccuracies in the learned diffusion prior, posterior sampling can become unstable and exploration can be misled.

Randomized Exploration In sequential decision making problems such as bandits and reinforcement learning (RL), randomized exploration often outperforms deterministic strategies by preventing early convergence to suboptimal actions (Jin et al., 2021; 2023). Among such methods, Thompson Sampling (TS) (Thompson, 1933) is a key approach for multi-armed bandits (Agrawal & Goyal, 2017), contextual bandits (Agrawal & Goyal, 2013), and RL (Osband et al., 2013). Unlike Upper Confidence Bound (UCB) algorithms, which rely on deterministic confidence intervals (Chu et al., 2011; Lattimore & Szepesvári, 2020), TS samples from posterior distributions, enabling robust exploration (Chapelle & Li, 2011). However, exact posterior sampling in TS is computationally intensive, particularly for non-conjugate priors. To address this, approximate sampling methods such as Langevin Monte Carlo (LMC) (Xu et al., 2022; Hsu et al., 2024), Stochastic gradient Langevin dynamics (SGLD) variants (Mazumdar et al., 2020; Zheng et al., 2024) and variational inference (Clavier et al., 2024) have been developed and applied to various problem settings, including multi-armed bandits with non-conjugate or highly nonlinear rewards, nonlinear contextual bandits and RL (Ishfaq et al., 2024a;b; Hsu et al., 2024). Perturb History Exploration (PHE) improves efficiency by perturbing historical data to approximate posterior sampling, making it applicable to complex reward distributions (Kveton et al., 2020; Ishfaq et al., 2021). Ensemble sampling keeps a small set of independently perturbed model replicas and selects actions using a randomly chosen replica (Lu & Van Roy, 2017; Qin et al., 2022; Janz et al., 2024; Lee & Oh, 2024).

B Additional Discussion and Algorithmic Details

This appendix complements Section 3 by providing additional motivation, intuition, and derivations for our diffusion posterior sampling framework. We first give an explanation of why diffusion models are suitable for posterior sampling with a learned prior to solve decision-making problems, and then provide detailed algorithmic interpretations for DLTS and DPSG.

B.1 Motivation and Mechanism of Diffusion Posterior Sampling

Why use diffusion models? TS requires drawing samples from the posterior $p(\boldsymbol{\theta} \mid \mathcal{H}_t)$ at every round t, necessitating a computationally sound method to combine a complex learned prior with likelihood terms. Diffusion models are a good fit because their iterative, L-step reverse process enables gradual and stable incorporation of likelihood guidance and randomized exploration. Specifi-

cally, Normalizing Flows (NFs) define distributions via a one-shot deterministic map (Papamakarios et al., 2021). While efficient for prior sampling, this structure does not naturally provide intermediate stochastic states at multiple noise levels for per-level conditioning. As a result, a straightforward use of NFs may require additional per-round inference (e.g., MCMC or variational inference) to incorporate the accumulating likelihood factors online, whereas diffusion models support conditional sampling by injecting likelihood drift throughout the reverse process equation 3.1. Furthermore, compared to score-only Energy-Based Models (EBMs) (Bond-Taylor et al., 2021), diffusion models can be more stable in practice. While EBMs allow sampling via MCMC in the original parameter space, this process can suffer from slow mixing on high-dimensional, multimodal distributions (Song & Ermon, 2019; Song et al., 2021).

The above discussion explains why an iterative sampler is useful for combining a learned multi-task prior with the likelihood induced by online bandit feedback. We next illustrate how this mechanism appears in our two algorithmic instantiations.

Mechanism of Diffusion Posterior Sampling Figure 4 illustrates three ways to sample the parameter $\boldsymbol{\theta}_0$ via a diffusion reverse process for contextual bandits. The left panel shows unconditional reverse diffusion (prior): starting from noise at level L, successive reverse transitions (blue arrows) move the particle cloud toward a structured manifold at level 0. However, this process does not use the bandit interaction history. The middle panel (DLTS) incorporates the interaction history $\mathcal{H}_t$ by warping the sequence of hyperplanes, so each reverse step jumps between adjacent history-conditioned levels, yielding a posterior sample path. The right panel (DPSG) keeps the hyperplanes unchanged (same geometry as the prior) but adds a guidance correction at each level: after each unconditional reverse step (blue arrows), a history-based guidance step adjusts particles before proceeding to the next level.

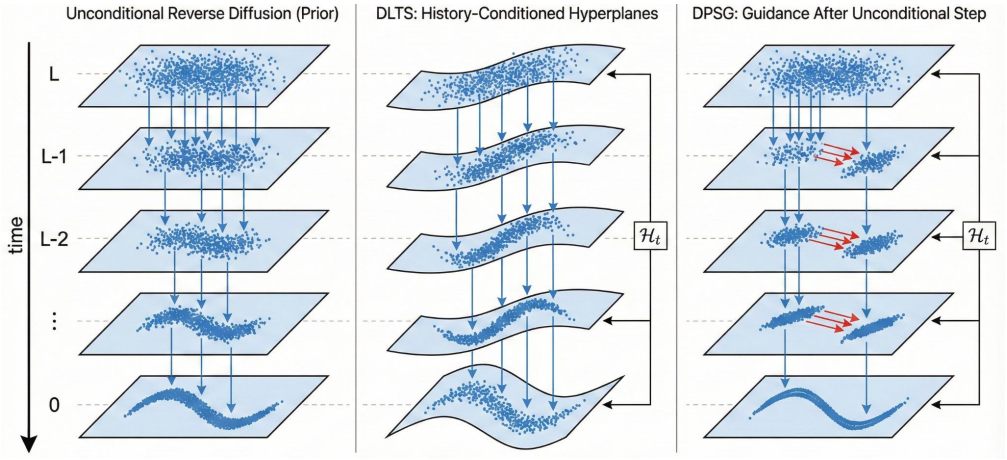


Figure 4: Diffusion Posterior Sampling for Contextual Bandits: Prior (left) vs DLTS (middle) vs DPSG (right). Blue arrow: reverse diffusion transition; Red arrow: guidance step (DPSG only).

This visualization highlights the main difference between DLTS and DPSG: DLTS builds history conditioning into each reverse transition, whereas DPSG first follows the learned prior dynamics and then applies a likelihood-guided correction. We now provide the detailed derivation and algorithmic interpretation of these two implementations.

B.2 Detailed Algorithm Interpretation for DLTS

We run DLTS for T rounds to solve the new bandit task $(N+1)$. At each time t, with a pretrained diffusion model for unconditional sampling, we modify the reverse diffusion update to perform conditional sampling by injecting the history at every reverse step. We implement this conditional

sampling by applying approximate sampling via Langevin Monte Carlo (LMC). After the reverse process ends, we take the final state $\boldsymbol{\theta}_0$ (i.e., $\widetilde{\boldsymbol{\theta}}_t$) as the parameter sample for arm selection.

Next, we provide a detailed derivation of the conditional sampling used in the reverse diffusion update in Algorithm 1. After training the diffusion denoiser $\varepsilon_{\boldsymbol{\phi}^*}(\cdot,\cdot)$ (see Appendix F for more details), we obtain the diffusion model parameter $(\boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell)_{\ell\in[L+1]}$[1] for unconditional sampling in each reverse step, which is formulated as follows,

$$p\big(\boldsymbol{\theta}_{\ell-1}|\boldsymbol{\theta}_\ell\big) = \mathcal{N}\big(\boldsymbol{\theta}_{\ell-1}; \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell\big), \tag{B.1}$$

where $\boldsymbol{\mu}_\ell = \frac{1}{\sqrt{\alpha_\ell}}\big(\boldsymbol{\theta}_\ell - \frac{1-\alpha_\ell}{\sqrt{1-\bar{\alpha}_\ell}}\varepsilon_{\boldsymbol{\phi}^*}(\boldsymbol{\theta}_\ell, \ell)\big)$, $\boldsymbol{\Sigma}_\ell = \frac{1-\bar{\alpha}_{\ell-1}}{1-\bar{\alpha}_\ell}\beta_\ell \mathbf{I}$. Note that in equation 3.2, we can factorize $p\big(\mathcal{H}_t|\boldsymbol{\theta}_{\ell-1}\big)$ as follows,

$$p\big(\mathcal{H}_t|\boldsymbol{\theta}_{\ell-1}\big) = \int_{\boldsymbol{\theta}_0} p\big(\mathcal{H}_t|\boldsymbol{\theta}_0, \boldsymbol{\theta}_{\ell-1}\big)p\big(\boldsymbol{\theta}_0|\boldsymbol{\theta}_{\ell-1}\big)d\boldsymbol{\theta}_0 = \int_{\boldsymbol{\theta}_0} p\big(\mathcal{H}_t|\boldsymbol{\theta}_0\big)p\big(\boldsymbol{\theta}_0|\boldsymbol{\theta}_{\ell-1}\big)d\boldsymbol{\theta}_0,$$

where the second equality comes from that $\mathcal{H}_t$ and $\boldsymbol{\theta}_{\ell-1}$ are conditionally independent on $\boldsymbol{\theta}_0$. To derive the conditional diffusion reverse update, we need to approximate the following term $\int_{\boldsymbol{\theta}_0} p(\mathcal{H}_t|\boldsymbol{\theta}_0)p(\boldsymbol{\theta}_0|\boldsymbol{\theta}_{\ell-1})d\boldsymbol{\theta}_0$. Based on Kveton et al. (2024), we approximate it by

$$\int_{\boldsymbol{\theta}_0} p\big(\mathcal{H}_t|\boldsymbol{\theta}_0\big)p\big(\boldsymbol{\theta}_0\big|\boldsymbol{\theta}_{\ell-1}\big)d\boldsymbol{\theta}_0 \approx p\big(\mathcal{H}_t|\boldsymbol{\theta}_{\ell-1}/\sqrt{\bar{\alpha}_{\ell-1}}\big). \tag{B.2}$$

The motivation of approximation in equation B.2 is as follows. Note that $\boldsymbol{\theta}_\ell = \sqrt{\bar{\alpha}_\ell}\boldsymbol{\theta}_0 + \sqrt{1-\bar{\alpha}_\ell}\widetilde{\boldsymbol{\xi}}_\ell$, where $\widetilde{\boldsymbol{\xi}}_\ell \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is standard Gaussian noise. Rearranging gives $\boldsymbol{\theta}_0 = (\boldsymbol{\theta}_\ell - \sqrt{1-\bar{\alpha}_\ell}\widetilde{\boldsymbol{\xi}}_\ell)/\sqrt{\bar{\alpha}_\ell}$, so $\boldsymbol{\theta}_0$ can be viewed as a random variable with mean $\boldsymbol{\theta}_\ell/\sqrt{\bar{\alpha}_\ell}$. Replacing $\boldsymbol{\theta}_0$ in the LHS of equation B.2 by this mean yields the approximation.

Note that at step t, given a pre-determined f, the nonlinear reward model is defined as

$$y_t = f(\boldsymbol{\theta}^*; \boldsymbol{x}_t) + \varepsilon_t.$$

When we assume Gaussian noise $\varepsilon_t \sim \mathcal{N}(0, \sigma^2)$, then the likelihood becomes

$$p\big(\mathcal{H}_t|\boldsymbol{\theta}\big) \propto \exp\Bigg(-\sigma^{-2}\sum_{i=1}^{t}\big(f(\boldsymbol{\theta}; \boldsymbol{x}_i) - y_i\big)^2\Bigg) = \exp(-L_t(\boldsymbol{\theta})), \tag{B.3}$$

where $L_t(\boldsymbol{\theta}) = \sigma^{-2}\sum_{i=1}^{t}(f(\boldsymbol{\theta}; \boldsymbol{x}_i) - y_i)^2$. Therefore, based on equation 3.2, equation B.2 and equation B.3, we approximate the posterior in each reverse step for conditional sampling as follows,

$$p\big(\boldsymbol{\theta}_{\ell-1}|\boldsymbol{\theta}_\ell, \mathcal{H}_t\big) \propto \exp\big(-L_t\big(\boldsymbol{\theta}_{\ell-1}/\sqrt{\bar{\alpha}_{\ell-1}}\big)\big) \cdot \mathcal{N}\big(\boldsymbol{\theta}_{\ell-1}; \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma}_\ell\big). \tag{B.4}$$

To sample from $p\big(\boldsymbol{\theta}_{\ell-1}|\boldsymbol{\theta}_\ell, \mathcal{H}_t\big)$, we apply Langevin Monte Carlo for approximate sampling. Specifically, at bandit round t and diffusion reverse step ℓ, we iteratively conduct the Langevin update,

$$\boldsymbol{\theta}_{\ell,k+1} = \boldsymbol{\theta}_{\ell,k} - \eta_\ell\Big[\nabla_{\boldsymbol{\theta}} L_t\big(\boldsymbol{\theta}_{\ell,k}/\sqrt{\bar{\alpha}_{\ell-1}}\big) + \boldsymbol{\Sigma}_\ell^{-1}\big(\boldsymbol{\theta}_{\ell,k} - \boldsymbol{\mu}_\ell\big)\Big] + \sqrt{2\eta_\ell\zeta_\ell^{-1}}\boldsymbol{\xi}_k, \tag{B.5}$$

where $\boldsymbol{\xi}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, η_ℓ is step-size, ζ_ℓ is temperature and equation B.5 is initialized at $\boldsymbol{\theta}_{\ell,0} = \boldsymbol{\mu}_\ell$ and after K_ℓ iterations return $\boldsymbol{\theta}_{\ell,K_\ell}$ to be the diffusion reverse process sample $\boldsymbol{\theta}_{\ell-1}$.

Specially, when we only conduct one LMC update, based on equation B.5, we have the reverse update

$$\boldsymbol{\theta}_{\ell-1} \leftarrow \Big(1 - \frac{(1-\bar{\alpha}_\ell)\eta_\ell}{(1-\bar{\alpha}_{\ell-1})\beta_\ell}\Big)\boldsymbol{\theta}_\ell + \frac{(1-\bar{\alpha}_\ell)\eta_\ell}{(1-\bar{\alpha}_{\ell-1})\beta_\ell}\boldsymbol{\mu}_\ell - \eta_\ell\nabla_{\boldsymbol{\theta}} L_t\big(\boldsymbol{\theta}_\ell/\sqrt{\bar{\alpha}_{\ell-1}}\big) + \sqrt{2\eta_\ell\zeta_\ell^{-1}}\boldsymbol{\xi}. \tag{B.6}$$

Note that equation B.6 aligns with our unified update equation 3.1. This confirms the rationality of our framework and the essence of diffusion-based posterior sampling. Intuitively, DLTS forms a conditional posterior by incorporating the history into the diffusion prior at each reverse step, then draws samples from it via approximate sampling.

[1] We define $(\boldsymbol{\mu}_{L+1}, \boldsymbol{\Sigma}_{L+1}) = (\mathbf{0}, \mathbf{I})$ for notation simplicity.

B.3 Detailed Algorithm Interpretation for DPSG

We run DPSG for T rounds to solve the new bandit task $(N+1)$. At each time t, after using a pretrained diffusion model for unconditional sampling, we then turn the reverse diffusion into conditional sampling by adding a likelihood drift at every reverse step: at level ℓ we form the Tweedie proxy $\hat{\boldsymbol{\theta}}_0$ from the current state and apply a single gradient update inside the reverse transition. The history interaction data enter as one likelihood-score step per level. After the reverse process ends, we take the final state $\boldsymbol{\theta}_0$ (i.e., $\widetilde{\boldsymbol{\theta}}_t$) as the parameter sample for arm selection.

To derive diffusion reverse update (Line 8 in Algorithm 2), we follow Chung et al. (2023) and decompose the conditional score into two terms,

$$\nabla_{\boldsymbol{\theta}} \log p_s(\boldsymbol{\theta}_s|y) = \underbrace{\nabla_{\boldsymbol{\theta}} \log p_s(\boldsymbol{\theta}_s)}_{\text{unconditional score}} + \underbrace{\nabla_{\boldsymbol{\theta}} \log p_s(y|\boldsymbol{\theta}_s)}_{\text{likelihood score}}.$$

For the first term, we can use pretrained diffusion model score network s_{ψ^*} to approximate the unconditional score. For the second term, we can use Tweedie's formula to provide a tractable approximation for $p_s(y|\boldsymbol{\theta}_s)$ such that one can use the surrogate function for approximate posterior sampling, which is formulated as,

$$\nabla_{\boldsymbol{\theta}} \log p_s(y|\boldsymbol{\theta}_s) \simeq \nabla_{\boldsymbol{\theta}} \log p_s\big(y|\hat{\boldsymbol{\theta}}_0\big), \tag{B.7}$$

where $\boldsymbol{\theta}$ is the conditional posterior mean of $\boldsymbol{\theta}_0$, which have the following close-form expression derived by Tweedie's formula (Lemma E.2), that is,

$$\hat{\boldsymbol{\theta}}_0 = \mathbb{E}_{\boldsymbol{\theta}_0 \sim p(\boldsymbol{\theta}_0|\boldsymbol{\theta}_s)}[\boldsymbol{\theta}_0] = \frac{1}{\sqrt{\bar{\alpha}_s}}\big(\boldsymbol{\theta}_s + (1-\bar{\alpha}_s)\nabla_{\boldsymbol{\theta}} \log p_s(\boldsymbol{\theta}_s)\big).$$

In practice, given learned score network s_{ψ^*}, we can further approximate the posterior mean as

$$\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_s, s) \simeq \frac{1}{\sqrt{\bar{\alpha}_s}}\big(\boldsymbol{\theta}_s + (1-\bar{\alpha}_s) s_{\psi^*}(\boldsymbol{\theta}_s, s)\big). \tag{B.8}$$

At timestep t, the nonlinear reward model is defined as $y_t = f(\boldsymbol{\theta}^*; \boldsymbol{x}_t) + \varepsilon_t$, where f is a predetermined nonlinear function. Same as equation B.3, when we assume $\varepsilon_t \sim \mathcal{N}(0, \sigma^2)$, then the likelihood becomes $p(\mathcal{H}_t|\boldsymbol{\theta}) \propto \exp(-\sigma^{-2}\sum_{i=1}^t (f(\boldsymbol{\theta}; \boldsymbol{x}_i) - y_i)^2) = \exp(-L_t(\boldsymbol{\theta}))$ where $L_t(\boldsymbol{\theta}) = \sigma^{-2}\sum_{i=1}^t (f(\boldsymbol{\theta}; \boldsymbol{x}_i) - y_i)^2$. Based on equation B.7 and equation B.8, then we can have the following tractable approximation of likelihood score,

$$\nabla_{\boldsymbol{\theta}} \log p_s(\mathcal{H}_t|\boldsymbol{\theta}_s) \simeq -\nabla_{\boldsymbol{\theta}} L_t\big(\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_s, s)\big).$$

Therefore, we can further have the approximation of the conditional score of reverse dynamics,

$$\nabla_{\boldsymbol{\theta}} \log p_s(\boldsymbol{\theta}_s|\mathcal{H}_t) \simeq s_{\psi^*}(\boldsymbol{\theta}_s, s) - \nabla_{\boldsymbol{\theta}} L_t\big(\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_s, s)\big). \tag{B.9}$$

In practice, to obtain a discrete-time algorithm we use DDPM ancestral sampling (Ho et al., 2020) to implement conditional sampling via the conditional score equation B.9. The reverse update is done by two steps. First, we use the unconditional score $s_{\psi^*}(\boldsymbol{\theta}_\ell, \ell)$ through Tweedie's estimate $\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_\ell, \ell)$ to form the Gaussian posterior mean, we then sample

$$\boldsymbol{\theta}'_{\ell-1} \leftarrow \frac{\sqrt{\alpha_\ell}(1-\bar{\alpha}_{\ell-1})}{1-\bar{\alpha}_\ell}\boldsymbol{\theta}_\ell + \frac{\sqrt{\bar{\alpha}_{\ell-1}}\beta_\ell}{1-\bar{\alpha}_\ell}\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_\ell, \ell) + \boldsymbol{z}_\ell, \quad \boldsymbol{z}_\ell \sim \mathcal{N}(\mathbf{0}, \beta_\ell \mathbf{I}), \tag{B.10}$$

Based on equation B.9, the second step is that we make this update conditional by subsequently adding one likelihood term. In practice, after sampling $\boldsymbol{\theta}'_{\ell-1}$, we take a gradient step using the likelihood L_t at the Tweedie estimate for guidance, that is

$$\boldsymbol{\theta}_{\ell-1} \leftarrow \boldsymbol{\theta}'_{\ell-1} - \eta_\ell \nabla_{\boldsymbol{\theta}} L_t\big(\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_\ell, \ell)\big). \tag{B.11}$$

When we merge equation B.10 and equation B.11 together, we obtain line 8 in Algorithm 2,

$$\boldsymbol{\theta}_{\ell-1} \leftarrow \frac{1}{\sqrt{\alpha_\ell}}\boldsymbol{\theta}_\ell + \frac{\beta_\ell}{\sqrt{\alpha_\ell}} s_{\psi^*}(\boldsymbol{\theta}_\ell, \ell) - \eta_\ell \nabla_{\boldsymbol{\theta}} L_t\big(\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_\ell, \ell)\big) + \boldsymbol{z}_\ell. \tag{B.12}$$

Note that equation B.12 aligns with our unified update equation 3.1. This confirms the rationality of our framework and the essence of diffusion-based posterior sampling. However, from the intuition perspective, DPSG is to use history information as guidance after unconditional update based on the diffusion prior. This is different from DLTS, which directly integrates history information into the unconditional update based on the diffusion prior to achieve conditional sampling.

B.4 Algorithm Interpretation for DPSG-MP

DPSG is simple and fast, but it relies on the likelihood-score at the Tweedie estimate. A recent work Xu et al. (2025) observes that DPS has the properties of high bias and low diversity, thus behaving like an implicit, but unstable, MAP estimator. Therefore, multi-step projection is used to solve this MAP optimization problem. Xu et al. (2025) proposed Diffusion Maximize a Posterior (DMAP), which keeps the standard reverse diffusion prior step and replaces the single likelihood drift with several small gradient-ascent steps on the log posterior at each noise level, projecting after each step onto a sphere around the unconditional mean to stay consistent with the diffusion dynamics.

Therefore, we introduce DPSG with Multi-step Projection (DPSG-MP) shown in Algorithm 3. Compared to Algorithm 2, it replaces the single correction with a short inner loop of gradient ascent on the likelihood score, and after each step projects the iterate onto the sphere where the reverse transition $p_{\psi^*}(\boldsymbol{\theta}_{\ell-1}|\boldsymbol{\theta}_\ell)$ concentrates around. This avoids the need for an accurate conditional score, reduces drift bias, and yields stable, data-consistent parameter estimates for arm selection.

Algorithm 3 DPSG with Multi-step Projection (DPSG-MP)

Input: score network $s_{\psi^*}(\cdot,\cdot)$, noise schedule $\{\beta_\ell\}$, trajectory history $\mathcal{H}_t$, learning rate $\{\eta_{\ell,k}\}$.

$\alpha_\ell \leftarrow 1-\beta_\ell$ and $\bar{\alpha}_\ell \leftarrow \prod_{j=1}^{\ell}\alpha_j$ for all ℓ.
for $t = 1, 2, \cdots, T$ **do**
  Receive contextual vector $\{\boldsymbol{x}_t(a)\}_{a\in\mathcal{A}}$.
  $\boldsymbol{\theta}_L \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.
  for $\ell = L, L-1, \cdots, 1$ **do**
    $\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_\ell, \ell) \leftarrow \frac{1}{\sqrt{\bar{\alpha}_\ell}}\big(\boldsymbol{\theta}_\ell + (1-\bar{\alpha}_\ell)s_{\psi^*}(\boldsymbol{\theta}_\ell, \ell)\big)$.
    $\boldsymbol{z}_\ell \sim \mathcal{N}(\mathbf{0}, \beta_\ell\mathbf{I})$, $r_\ell \leftarrow \|\boldsymbol{z}_\ell\|_2$.
    $\boldsymbol{m}_\ell \leftarrow \frac{\sqrt{\alpha_\ell}(1-\bar{\alpha}_{\ell-1})}{1-\bar{\alpha}_\ell}\boldsymbol{\theta}_\ell + \frac{\sqrt{\bar{\alpha}_{\ell-1}}\beta_\ell}{1-\bar{\alpha}_\ell}\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_\ell, \ell)$.
    $\boldsymbol{\theta}_{\ell-1,0} \leftarrow \boldsymbol{m}_\ell + \boldsymbol{z}_\ell$.
    for $k = 1, 2, \cdots, K_\ell$ **do**
      $\boldsymbol{\theta}'_{\ell-1,k} \leftarrow \boldsymbol{\theta}_{\ell-1,k-1} - \eta_{\ell,k}\nabla_{\boldsymbol{\theta}} L_t\big(\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_\ell, \ell)\big)$.
      $\boldsymbol{\theta}_{\ell-1,k} \leftarrow \boldsymbol{m}_\ell + r_\ell \frac{\boldsymbol{\theta}'_{\ell-1,k} - \boldsymbol{m}_\ell}{\|\boldsymbol{\theta}'_{\ell-1,k} - \boldsymbol{m}_\ell\|_2}$.
    end for
    $\boldsymbol{\theta}_{\ell-1} \leftarrow \boldsymbol{\theta}_{\ell-1,K_\ell}$.
  end for
  Let $\widetilde{\boldsymbol{\theta}}_t \leftarrow \boldsymbol{\theta}_0$, take action $\boldsymbol{x}_t = \arg\max_{\boldsymbol{x}_t(a)} f\big(\widetilde{\boldsymbol{\theta}}_t; \boldsymbol{x}_t(a)\big)$.
  Receive reward y_t and update history $\mathcal{H}_{t+1} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^t$.
end for

C Proof in Section 4

Proof of Theorem 4.4. We show that, under the ideal ODTS assumptions, the ODTS marginal posterior over the clean parameter is the same as the OTS posterior.

Let the diffusion-path variable be $\boldsymbol{\theta}_{0:L} = (\boldsymbol{\theta}_0, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_L)$, where $\boldsymbol{\theta}_0$ is the clean parameter used for decisions and $\boldsymbol{\theta}_L$ is the noisiest level. The diffusion prior has the Markov factorization

$$p(\boldsymbol{\theta}_{0:L}) = p(\boldsymbol{\theta}_L)\prod_{\ell=1}^{L} p(\boldsymbol{\theta}_{\ell-1}|\boldsymbol{\theta}_\ell), \tag{C.1}$$

so that its marginal posterior is as follows,

$$p_0(\boldsymbol{\theta}_0) = \int p(\boldsymbol{\theta}_{0:L})\, d\boldsymbol{\theta}_{1:L}, \tag{C.2}$$

which is the true prior used by OTS according to the condition. Note that the history $\mathcal{H}_t$ depends only on $\boldsymbol{\theta}_0$, then we have

$$p(\mathcal{H}_t|\boldsymbol{\theta}_{0:L}) = p(\mathcal{H}_t|\boldsymbol{\theta}_0). \tag{C.3}$$

ODTS is assumed ideal: (i) the learned score/denoiser equals the true score so that equation C.1 holds for the implemented prior, and (ii) each per-level conditional reverse transition kernel is implemented exactly, thus representing the true joint posterior over the path.

By Bayes' rule and equation C.3, the joint posterior over the path satisfies

$$\widetilde{p}_t(\boldsymbol{\theta}_{0:L}) \equiv p(\boldsymbol{\theta}_{0:L}|\mathcal{H}_t) \propto p(\mathcal{H}_t|\boldsymbol{\theta}_0)p(\boldsymbol{\theta}_{0:L}). \tag{C.4}$$

Integrating equation C.4 over the noisy layers $\boldsymbol{\theta}_{1:L}$ and using equation C.2 gives

$$\begin{aligned}\int p(\boldsymbol{\theta}_{0:L}|\mathcal{H}_t)d\boldsymbol{\theta}_{1:L} &= \frac{1}{p(\mathcal{H}_t)}\int p(\mathcal{H}_t|\boldsymbol{\theta}_0)p(\boldsymbol{\theta}_{0:L})d\boldsymbol{\theta}_{1:L} \\ &= \frac{1}{p(\mathcal{H}_t)}p(\mathcal{H}_t|\boldsymbol{\theta}_0)\underbrace{\int p(\boldsymbol{\theta}_{0:L})d\boldsymbol{\theta}_{1:L}}_{=p_0(\boldsymbol{\theta}_0)} \\ &= \frac{p(\mathcal{H}_t|\boldsymbol{\theta}_0)p_0(\boldsymbol{\theta}_0)}{p(\mathcal{H}_t)} = p(\boldsymbol{\theta}_0|\mathcal{H}_t).\end{aligned} \tag{C.5}$$

Note that $\widetilde{p}_t(\boldsymbol{\theta}) = \int \widetilde{p}_t(\boldsymbol{\theta}_{0:L})d\boldsymbol{\theta}_{1:L} = \int p(\boldsymbol{\theta}_{0:L}|\mathcal{H}_t)d\boldsymbol{\theta}_{1:L}$, then we have

$$\widetilde{p}_t(\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\mathcal{H}_t) \equiv p_t(\boldsymbol{\theta}). \tag{C.6}$$

Therefore the ODTS posterior marginal $\widetilde{p}_t(\boldsymbol{\theta})$ and the OTS posterior $p_t(\boldsymbol{\theta})$ are the same. Note that both oracle algorithms use greedy policy to select arm, thus share the same arm selection distribution. □

D Instantiation Under Specific Settings

In this section, we describe the bandits settings where we instantiate our proposed algorithms. The general frameworks for DLTS and DPSG can be readily adapted to various specific models by defining the reward function $f(\boldsymbol{\theta}; \boldsymbol{x})$ and the corresponding loss function $L_t(\boldsymbol{\theta})$. The gradient of this loss, $\nabla_{\boldsymbol{\theta}} L_t(\boldsymbol{\theta})$, is the crucial model-dependent component that is plugged into the reverse diffusion update rules for both DLTS and DPSG. We now provide the concrete formulations for three widely-studied settings: linear, generalized linear, and neural contextual bandits.

Linear Bandit We first consider the linear contextual bandit settings. In linear contextual bandits, the reward is given by a linear function of the context and an unknown parameter $\boldsymbol{\theta}^* \in \mathbb{R}^d$ plus some noise, that is, $y_t = \boldsymbol{\theta}^{*\top}\boldsymbol{x}_t + \varepsilon_t$, where $\boldsymbol{x}_t \in \mathbb{R}^d$ is the context at time t and ε_t is the noise term. We assume that ε_t is i.i.d. Gaussian noise with mean zero and variance σ^2. The goal of the agent is to learn the unknown parameter $\boldsymbol{\theta}^*$ and select actions that maximize the cumulative reward over time. The reward function is defined as follows

$$f(\boldsymbol{\theta}; \boldsymbol{x}) = \boldsymbol{\theta}^\top \boldsymbol{x}.$$

In our proposed algorithms, we define the loss function over the observed history $\mathcal{H}_t$ as follows

$$L_t(\boldsymbol{\theta}) = \sigma^{-2}\sum_{i=1}^{t}(\boldsymbol{\theta}^\top \boldsymbol{x}_i - y_i)^2,$$

where σ^2 is the variance, $\boldsymbol{x}_i$ is the context at time i and y_i is the corresponding reward. For the linear model, the crucial gradient term $\nabla_{\boldsymbol{\theta}} L_t(\boldsymbol{\theta})$ required for our diffusion updates has a convenient closed-form expression. It can be computed as:

$$\nabla_{\boldsymbol{\theta}} L_t(\boldsymbol{\theta}) = 2\sigma^{-2}\sum_{i=1}^{t}(\boldsymbol{\theta}^\top \boldsymbol{x}_i - y_i)\boldsymbol{x}_i.$$

For DLTS, the Langevin Monte Carlo update from equation B.6, which is iterated within each reverse diffusion step ℓ. We have

$$\boldsymbol{\theta}_{\ell-1} \leftarrow \left(1 - \frac{(1-\bar{\alpha}_\ell)\eta_\ell}{(1-\bar{\alpha}_{\ell-1})\beta_\ell}\right)\boldsymbol{\theta}_\ell + \frac{(1-\bar{\alpha}_\ell)\eta_\ell}{(1-\bar{\alpha}_{\ell-1})\beta_\ell}\boldsymbol{\mu}_\ell - 2\eta_\ell\sigma^{-2}\sum_{i=1}^{t}\left(\frac{\boldsymbol{\theta}_\ell^\top \boldsymbol{x}_i}{\sqrt{\bar{\alpha}_{\ell-1}}} - y_i\right)\boldsymbol{x}_i + \sqrt{2\eta_\ell\zeta_\ell^{-1}}\boldsymbol{\xi}.$$

For DPSG, the reverse update from Line 8 in Algorithm 2 is instantiated as:

$$\boldsymbol{\theta}_{\ell-1} \leftarrow \frac{1}{\sqrt{\alpha_\ell}}\boldsymbol{\theta}_\ell + \frac{\beta_\ell}{\sqrt{\alpha_\ell}}s_{\boldsymbol{\psi}^*}(\boldsymbol{\theta}_\ell, \ell) - 2\eta_\ell\sigma^{-2}\sum_{i=1}^{t}\left(\left(\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_\ell, \ell)\right)^\top \boldsymbol{x}_i - y_i\right)\boldsymbol{x}_i + \boldsymbol{z}_\ell.$$

Generalized Linear Bandit Then we extend our proposed algorithms to generalized linear bandit settings. In generalized linear bandits, the reward is given by a generalized linear function of the context and an unknown parameter $\boldsymbol{\theta}^* \in \mathbb{R}^d$ plus some noise, that is, $y_t = g(\boldsymbol{\theta}^{*\top}\boldsymbol{x}_t) + \varepsilon_t$, where $\boldsymbol{x}_t \in \mathbb{R}^d$ is the context at time t, g is a known link function and ε_t is the noise term. We assume that ε_t is i.i.d. Gaussian noise with mean zero and variance σ^2. The link function g is a known function that maps the linear combination of the context and parameter to the expected reward. We have the following formulations for the reward function

$$f(\boldsymbol{\theta}; \boldsymbol{x}) = g(\boldsymbol{\theta}^\top \boldsymbol{x}).$$

The goal of the agent is to learn the unknown parameter $\boldsymbol{\theta}^*$ and select actions that maximize the cumulative reward over time. In our proposed algorithms, we define the loss function over the observed history $\mathcal{H}_t$ as follows

$$L_t(\boldsymbol{\theta}) = \sigma^{-2} \sum_{i=1}^{t} (g(\boldsymbol{\theta}^\top \boldsymbol{x}_i) - y_i)^2,$$

where σ^2 is the variance, $\boldsymbol{x}_i$ is the context at time i and y_i is the corresponding reward. We use the logistic function as the example in our experiments. The logistic function is defined as $g(z) = 1/(1 + e^{-z})$. For the gradient, we have:

$$\nabla_{\boldsymbol{\theta}} L_t(\boldsymbol{\theta}) = 2\sigma^{-2} \sum_{i=1}^{t} (g(\boldsymbol{\theta}^\top \boldsymbol{x}_i) - y_i) g'(\boldsymbol{\theta}^\top \boldsymbol{x}_i) \boldsymbol{x}_i,$$

where $g'(\cdot)$ is the derivative of the link function with respect to its input.

For DLTS, the single-step Langevin update equation B.6 becomes:

$$\begin{aligned}\boldsymbol{\theta}_{\ell-1} \leftarrow & \left(1 - \frac{(1-\bar{\alpha}_\ell)\eta_\ell}{(1-\bar{\alpha}_{\ell-1})\beta_\ell}\right)\boldsymbol{\theta}_\ell + \frac{(1-\bar{\alpha}_\ell)\eta_\ell}{(1-\bar{\alpha}_{\ell-1})\beta_\ell}\boldsymbol{\mu}_\ell \\ & - 2\eta_\ell \sigma^{-2} \sum_{i=1}^{t} \left(g\left(\frac{\boldsymbol{\theta}_\ell^\top \boldsymbol{x}_i}{\sqrt{\bar{\alpha}_{\ell-1}}}\right) - y_i\right) g'\left(\frac{\boldsymbol{\theta}_\ell^\top \boldsymbol{x}_i}{\sqrt{\bar{\alpha}_{\ell-1}}}\right) \boldsymbol{x}_i + \sqrt{2\eta_\ell \zeta_\ell^{-1}}\boldsymbol{\xi}.\end{aligned}$$

For DPSG, the reverse update (Line 8 in Algorithm 2) is instantiated as:

$$\begin{aligned}\boldsymbol{\theta}_{\ell-1} \leftarrow & \frac{1}{\sqrt{\alpha_\ell}}\boldsymbol{\theta}_\ell + \frac{\beta_\ell}{\sqrt{\alpha_\ell}} s_{\boldsymbol{\psi}^*}(\boldsymbol{\theta}_\ell, \ell) \\ & - 2\eta_\ell \sigma^{-2} \sum_{i=1}^{t} \left(g\left(\left(\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_\ell, \ell)\right)^\top \boldsymbol{x}_i\right) - y_i\right) g'\left(\left(\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_\ell, \ell)\right)^\top \boldsymbol{x}_i\right) \boldsymbol{x}_i + \boldsymbol{z}_\ell.\end{aligned}$$

Neural Bandit Finally, we consider the neural contextual bandit settings. In neural contextual bandits, the reward is given by a neural network function of the context and an unknown parameter $\boldsymbol{\theta}^* \in \mathbb{R}^p$ plus some noise, that is, $y_t = f(\boldsymbol{\theta}^*; \boldsymbol{x}_t) + \varepsilon_t$, where $\boldsymbol{x}_t \in \mathbb{R}^d$ is the context at time t and ε_t is the noise term. We assume that ε_t is i.i.d. Gaussian noise with mean zero and variance σ^2. The goal of the agent is to learn the unknown parameter $\boldsymbol{\theta}^*$ and select actions that maximize the cumulative reward over time. The reward function is defined as follows

$$f(\boldsymbol{\theta}; \boldsymbol{x}) = \sqrt{m}\mathbf{W}_L \sigma(\mathbf{W}_{L-1}\sigma(\cdots \sigma(\mathbf{W}_1 \mathbf{x}))),$$

where $\sigma(x) = \max\{x, 0\}$ is the rectified linear unit (ReLU) activation function, $\mathbf{W}_1 \in \mathbb{R}^{m\times d}$, $\mathbf{W}_i \in \mathbb{R}^{m\times m}, 2 \le i \le L-1, \mathbf{W}_L \in \mathbb{R}^{m\times 1}$, and $\boldsymbol{\theta} = \left[\mathrm{vec}(\mathbf{W}_1)^\top, \ldots, \mathrm{vec}(\mathbf{W}_L)^\top\right]^\top \in \mathbb{R}^p$ with $p = m + md + m^2(L-2)$. In our proposed algorithms, we define the loss function over the observed history $\mathcal{H}_t$ as follows

$$L_t(\boldsymbol{\theta}) = \sigma^{-2} \sum_{i=1}^{t} (f(\boldsymbol{\theta}; \boldsymbol{x}_i) - y_i)^2,$$

where σ^2 is the variance, $\boldsymbol{x}_i$ is the context at time i and y_i is the corresponding reward. The gradient of this loss function is required for our diffusion updates. We obtain:

$$\nabla_{\boldsymbol{\theta}} L_t(\boldsymbol{\theta}) = 2\sigma^{-2} \sum_{i=1}^{t} (f(\boldsymbol{\theta}; \boldsymbol{x}_i) - y_i) \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}; \boldsymbol{x}_i).$$

For DLTS, the single-step Langevin update equation B.6 becomes:

$$\boldsymbol{\theta}_{\ell-1} \leftarrow \left(1 - \frac{(1-\bar{\alpha}_\ell)\eta_\ell}{(1-\bar{\alpha}_{\ell-1})\beta_\ell}\right)\boldsymbol{\theta}_\ell + \frac{(1-\bar{\alpha}_\ell)\eta_\ell}{(1-\bar{\alpha}_{\ell-1})\beta_\ell}\boldsymbol{\mu}_\ell - 2\eta_\ell\sigma^{-2}\sum_{i=1}^{t}\left(f\left(\frac{\boldsymbol{\theta}_\ell}{\sqrt{\bar{\alpha}_{\ell-1}}}; \boldsymbol{x}_i\right) - y_i\right)\nabla_{\boldsymbol{\theta}} f\left(\frac{\boldsymbol{\theta}_\ell}{\sqrt{\bar{\alpha}_{\ell-1}}}; \boldsymbol{x}_i\right) + \sqrt{2\eta_\ell\zeta_\ell^{-1}}\boldsymbol{\xi}.$$

For DPSG, the reverse update (Line 8 in Algorithm 2) is instantiated as:

$$\boldsymbol{\theta}_{\ell-1} \leftarrow \frac{1}{\sqrt{\alpha_\ell}}\boldsymbol{\theta}_\ell + \frac{\beta_\ell}{\sqrt{\alpha_\ell}} s_{\boldsymbol{\psi}^*}(\boldsymbol{\theta}_\ell, \ell) - 2\eta_\ell\sigma^{-2}\sum_{i=1}^{t}\left(f\left(\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_\ell, \ell); \boldsymbol{x}_i\right) - y_i\right)\nabla_{\boldsymbol{\theta}} f\left(\hat{\boldsymbol{\theta}}_0(\boldsymbol{\theta}_\ell, \ell); \boldsymbol{x}_i\right) + \boldsymbol{z}_\ell.$$

E Auxiliary Lemmas

The following lemma provides the conditional probability distributions for diffusion reverse process based on history $\mathcal{H}_t = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{t-1}$ at timestep t.

Lemma E.1 (Kveton et al. (2024))**.** For probability measure p over the reverse process, we have

$$\begin{aligned}
p(\boldsymbol{\theta}_L|\mathcal{H}_t) &\propto \left(\int_{\boldsymbol{\theta}_0} p(\mathcal{H}_t|\boldsymbol{\theta}_0)p(\boldsymbol{\theta}_0|\boldsymbol{\theta}_L)d\boldsymbol{\theta}_0\right)\cdot p(\boldsymbol{\theta}_L),\\
p(\boldsymbol{\theta}_{\ell-1}|\boldsymbol{\theta}_\ell, \mathcal{H}_t) &\propto \left(\int_{\boldsymbol{\theta}_0} p(\mathcal{H}_t|\boldsymbol{\theta}_0)p(\boldsymbol{\theta}_0|\boldsymbol{\theta}_{\ell-1})d\boldsymbol{\theta}_0\right)\cdot p(\boldsymbol{\theta}_{\ell-1}|\boldsymbol{\theta}_\ell), \quad \text{for } \ell = 2, \ldots, L,\\
p(\boldsymbol{\theta}_0|\boldsymbol{\theta}_1, \mathcal{H}_t) &\propto p(\mathcal{H}_t|\boldsymbol{\theta}_0)\cdot p(\boldsymbol{\theta}_0|\boldsymbol{\theta}_1).
\end{aligned}$$

Lemma E.2 (Tweedie's formula (Efron, 2011))**.** Let $p(\boldsymbol{y}|\boldsymbol{\eta})$ belong to the exponential family distribution $p(\boldsymbol{y}|\boldsymbol{\eta}) = p_0(\boldsymbol{y})\exp\left(\boldsymbol{\eta}^\top T(\boldsymbol{y}) - \varphi(\boldsymbol{\eta})\right)$, where $\boldsymbol{\eta}$ is the canonical vector of the family, $T(\boldsymbol{y})$ is some function of $\boldsymbol{y}$, and $\varphi(\boldsymbol{\eta})$ is the cumulant generation function which normalizes the density, and $p_0(\boldsymbol{y})$ is the density up to the scale factor when $\boldsymbol{\eta} = \mathbf{0}$. Then, the posterior mean $\hat{\boldsymbol{\eta}} := \mathbb{E}[\boldsymbol{\eta}|\boldsymbol{y}]$ should satisfy

$$\left(\nabla_{\boldsymbol{y}} T(\boldsymbol{y})\right)^\top \hat{\boldsymbol{\eta}} = \nabla_{\boldsymbol{y}} \log p(\boldsymbol{y}) - \nabla_{\boldsymbol{y}} \log p_0(\boldsymbol{y}).$$

F Diffusion Model Training

In this section, we introduce the training of diffusion model for bandit with nonlinear reward model, which is shown in Algorithm 4. The training process is a standard DDPM training process (Ho et al., 2020).

Algorithm 4 Diffusion Model Training

Input: Training dataset $\mathcal{D} = \{\boldsymbol{\theta}^{(i)}\}_{i=1}^{N}$, total diffusion steps L, noise schedule $\{\beta_\ell\}_{\ell=1}^{L}$, learning rate η, mini–batch size B, epochs E.

$\alpha_\ell \leftarrow 1 - \beta_\ell$ and $\bar{\alpha}_\ell \leftarrow \prod_{j=1}^{\ell} \alpha_j$ for all ℓ.
Initialization: denoiser parameters $\boldsymbol{\phi} \leftarrow$ `randn()`.
for epoch $= 1, \ldots, E$ **do**
  for all mini–batches $\mathcal{B} \subset \mathcal{D}$ of size B **do**
    Sample time indices $\ell_1, \ldots, \ell_B \sim \text{Unif}\{1, \ldots, L\}$.
    Sample noises $\varepsilon_1, \ldots, \varepsilon_B \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.
    for $b = 1, \ldots, B$ **do**
      $X_0^{(b)} \leftarrow \boldsymbol{\theta}^{(b)}$.
      $X_{\ell_b}^{(b)} \leftarrow \sqrt{\bar{\alpha}_{\ell_b}} X_0^{(b)} + \sqrt{1 - \bar{\alpha}_{\ell_b}} \varepsilon_b$. {forward diffusion}
      $\hat{\varepsilon}_b \leftarrow \varepsilon_{\boldsymbol{\phi}}\big(X_{\ell_b}^{(b)}, \ell_b\big)$. {network prediction}
    end for
    $\mathcal{L} \leftarrow \dfrac{1}{B} \displaystyle\sum_{b=1}^{B} \big\|\hat{\varepsilon}_b - \varepsilon_b\big\|_2^2$.
    Gradient decent: $\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} - \eta \nabla_{\boldsymbol{\phi}} \mathcal{L}$.
  end for
end for

Output: Trained diffusion model denoiser $\varepsilon_{\boldsymbol{\phi}}(\cdot, \cdot)$.

G Experiment Details

In this section, we provide details about our experiment setups and implementations.

G.1 Implementation Details

Diffusion Prior Implementation Our diffusion prior is implemented within a Denoising Diffusion Probabilistic Model (DDPM) framework, using the standard ϵ-prediction parameterization. The core of this model is a denoiser network responsible for predicting the noise at each diffusion step. We implement this denoiser using a Multi-Layer Perceptron (MLP) conditioned on a sinusoidal time embedding. The key architectural details of the MLP used at each diffusion step are summarized in Table 2.

Table 2: MLP Denoiser Configuration.

Parameter	Value
Hidden Layers	(32, 32)
Activation	ReLU
Optimizer	Adam

Hyperparameters We provide the hyperparameters for our proposed algorithm in Table 3 and Table 4.

G.2 Prior Distribution

The ‘cross’ problem is generated from a mixture of two highly correlated 2D Gaussian distributions, creating a distinct cross shape. The ‘rays’ distribution is formed using rejection sampling, where accepted points are constrained to lie close to the four cardinal axes, resembling rays emanating from the origin. The ‘triangles’ distribution, also generated via rejection sampling, consists of points within two triangular regions. The ‘swirl’ distribution is generated parametrically, creating a spiral pattern with added noise. The ‘H’ distribution uses rejection sampling to accept points within three

Table 3: Hyperparameters of DLTS

Reward Model	Prior Distribution	Update Steps	Step size	Noise Scale
Cosine	cross	1	0.05	0.005
	rays	10	0.01	0.005
	triangles	1	0.005	0.05
	swirl	1	0.01	0.01
	H	1	0.05	0.01
	corners	1	0.05	0.05
Quadratic	cross	1	0.01	0.005
	rays	10	0.05	0.005
	triangles	10	0.05	0.01
	swirl	1	0.05	0.005
	H	1	0.005	0.1
	corners	10	0.01	0.05
Sigmoid-gated	cross	1	0.05	0.05
	rays	1	0.05	0.05
	triangles	10	0.05	0.005
	swirl	10	0.01	0.1
	H	1	0.1	0.01
	corners	1	0.05	0.1

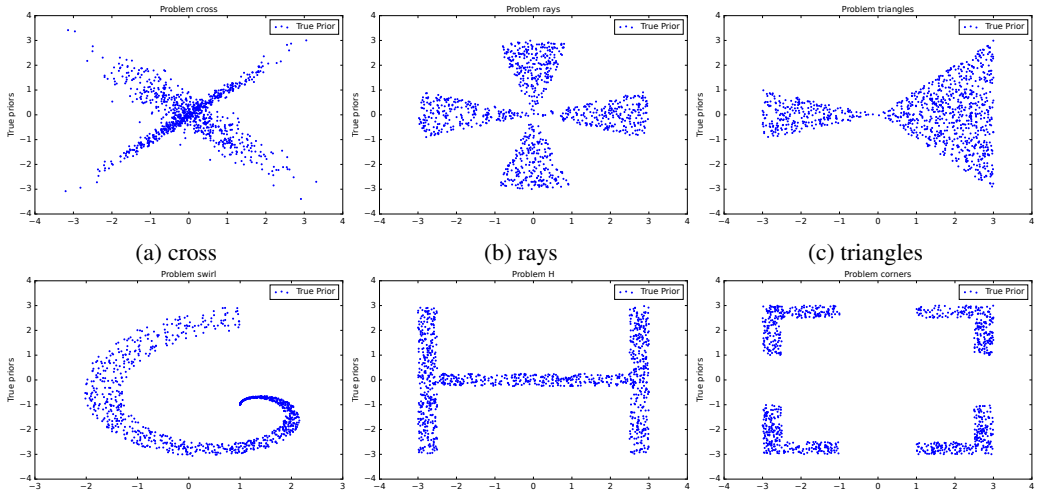
Table 4: Hyperparameters of DPSG-MP

Reward Model	Prior Distribution	Update Steps	Step Size
Cosine	cross	1	0.05
	rays	10	0.005
	triangles	10	0.01
	swirl	10	0.005
	H	1	0.05
	corners	1	0.05
Quadratic	cross	10	0.01
	rays	1	0.05
	triangles	1	0.1
	swirl	10	0.05
	H	10	0.005
	corners	10	0.001
Sigmoid-gated	cross	1	0.01
	rays	10	0.01
	triangles	10	0.1
	swirl	10	0.01
	H	10	0.05
	corners	10	0.05

rectangular regions that form the letter ‘H’. Finally, the ‘corners’ distribution consists of points sampled from four distinct rectangular areas located in the corners of the sampling domain.

H Additional Experimental Results

In this section, we present additional experimental results to complement those in Section 5.



(a) cross (b) rays (c) triangles

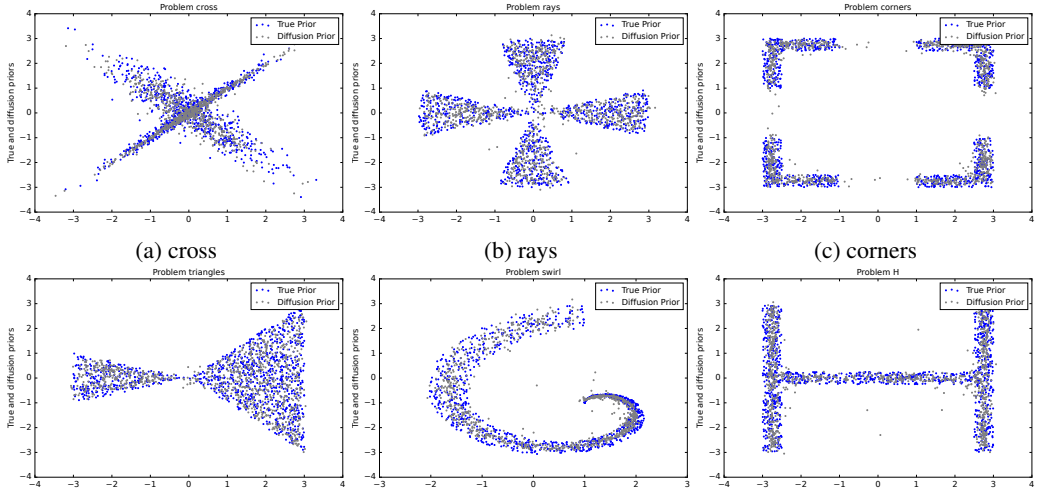
(d) swirl (e) H (f) corners

Figure 5: Prior Distribution. Each panel visualizes a prior over $\theta \in \mathbb{R}^2$. For each prior, we draw $N = 10000$ parameter vectors; 80% are used as training tasks (previously seen tasks) and 20% as test tasks (newly encountered tasks). These parameters determine the true reward functions in our simulated bandit environments.

H.1 Additional Results for Linear and Sigmoid-Gated Bandits

We provide learned-prior visualizations for all six prior distributions in the linear setting in Figure 6. The corresponding visualizations for the sigmoid-gated nonlinear setting are shown in Figure 8.

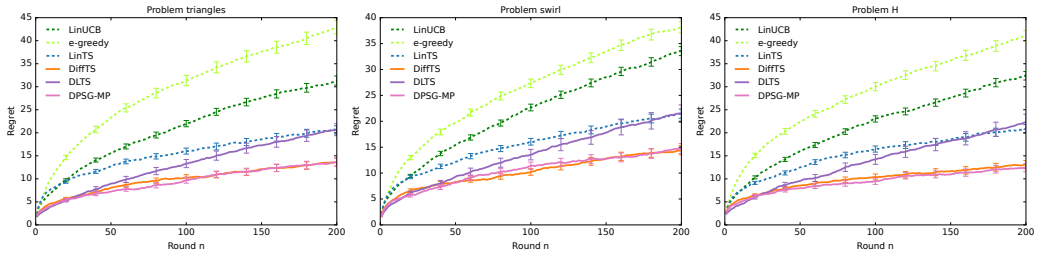
For the regret evaluation, the main text reports results for 'cross', 'rays', and 'corners'. Here, we provide the remaining results for 'triangles', 'swirl', and 'H' in the linear setting in Figure 7 and in the sigmoid-gated nonlinear setting in Figure 9.



(a) cross (b) rays (c) corners

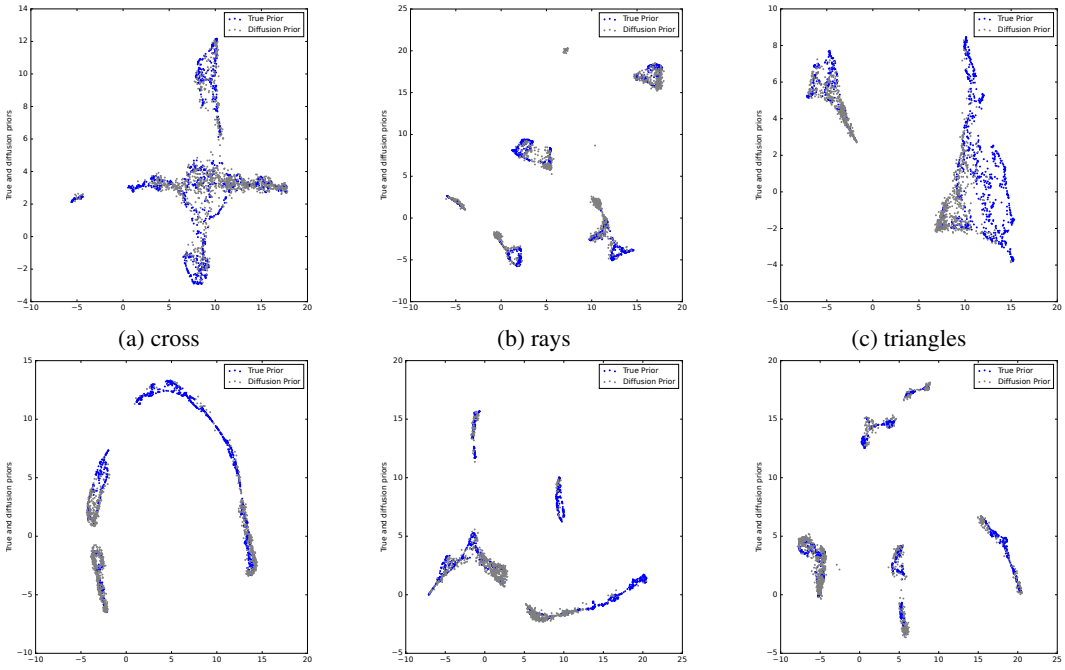
(d) triangles (e) swirl (f) H

Figure 6: Visualization of the learned diffusion prior versus the true prior. We demonstrate all six priors. For each figure, samples from our trained model (in grey) are overlaid on the ground-truth samples (in blue).



(a) triangles (b) swirl (c) H

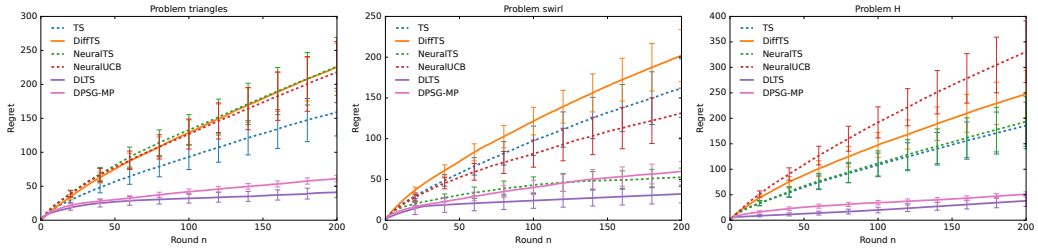
Figure 7: Performance of all algorithms on the linear contextual bandit tasks for the three prior distributions. Our proposed methods achieve cumulative regret comparable to LinTS and DiffTS.



(a) cross (b) rays (c) triangles

(d) swirl (e) H (f) corners

Figure 8: Visualization of the learned diffusion prior versus the true prior under the sigmoid-gated reward model. We use UMAP to project the high-dimensional MLP parameters into two dimensions. Samples from the learned diffusion prior (grey) are overlaid on the true prior samples (blue).



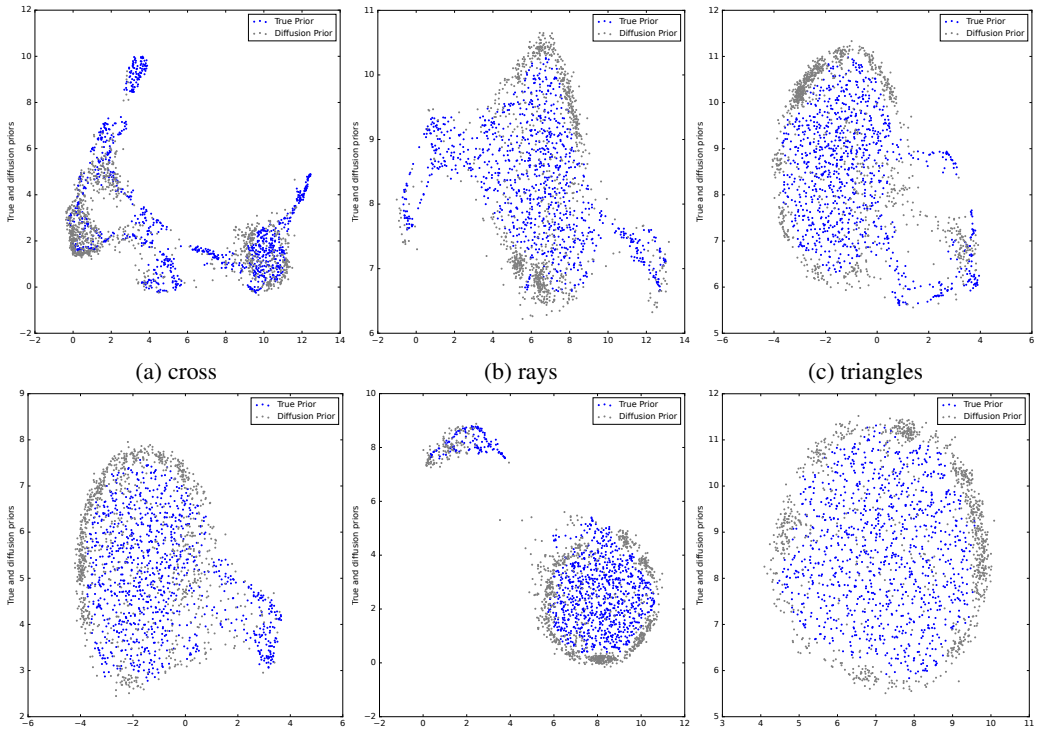
(a) triangles (b) swirl (c) H

Figure 9: Performance of all algorithms on the sigmoid-gated nonlinear bandit tasks for the three prior distributions. Our proposed methods outperform strong baselines like NeuralTS and NeuralUCB, achieving the lowest cumulative regret.

H.2 Results for Cosine and Quadratic Nonlinear Bandits

In the cosine reward setting, our proposed algorithms demonstrate strong performance, as shown in Figure 11. Both DLTS and the stabilized DPSG-MP achieve cumulative regret comparable to the state-of-the-art NeuralTS and NeuralUCB baselines. This result validates that a diffusion prior over neural network weights can effectively guide exploration in a complex, nonlinear environment. The reported results show the mean cumulative regret and standard error, averaged across 64 new tasks over 8 independent trials.

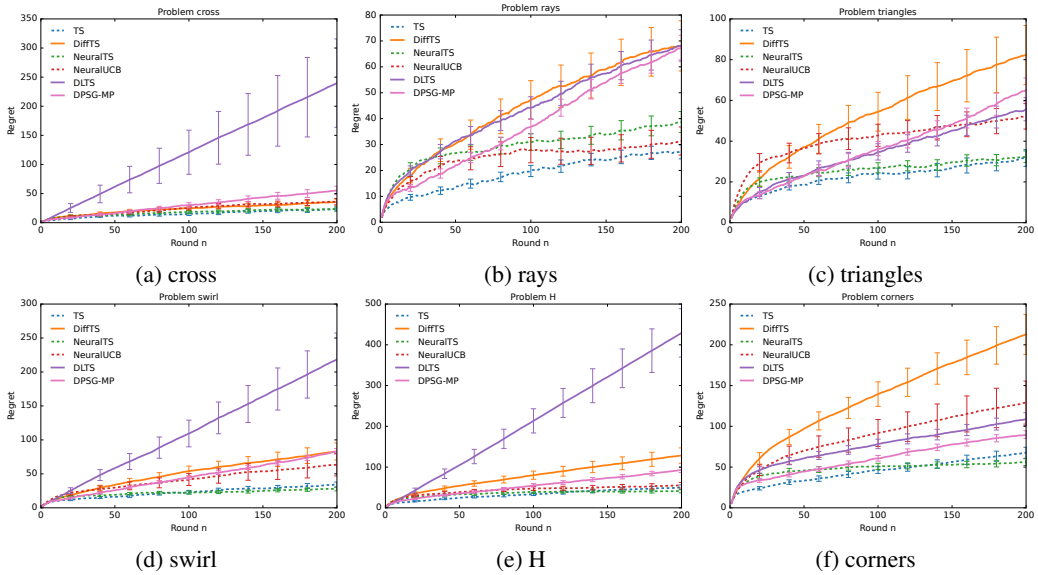
Similarly, for the quadratic reward model in Figure 13, our methods perform competitively against the strong neural baselines. Notably, for the challenging 'corners' prior, our algorithms outperform both NeuralTS and NeuralUCB. This highlights a key advantage of our approach: when the underlying parameter distribution has a complex structure, the guidance from the learned diffusion prior becomes particularly beneficial, leading to more efficient exploration and lower regret. The evaluation protocol remains the same, with results averaged over 64 tasks across 8 independent trials.



(a) cross (b) rays (c) triangles

(d) swirl (e) H (f) corners

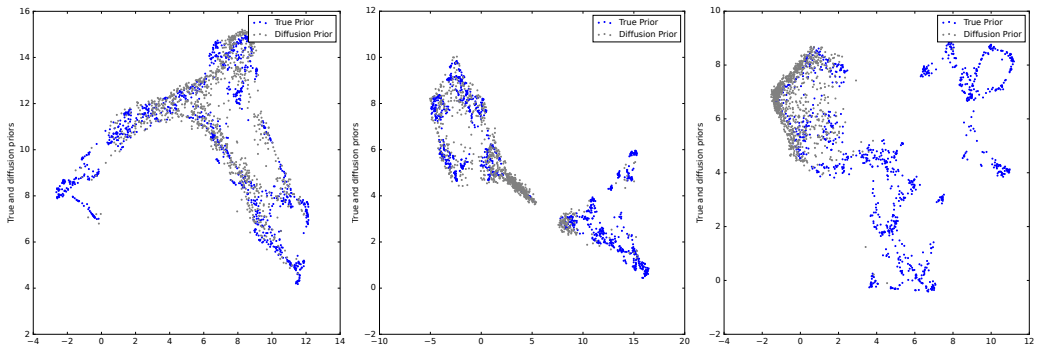
Figure 10: Visualization of the learned diffusion prior versus the true prior (under cosine reward model). The true prior is the MLP parameters learned from the previous task. Samples from our trained model (grey) are overlaid on the ground-truth samples (blue).



(a) cross (b) rays (c) triangles

(d) swirl (e) H (f) corners

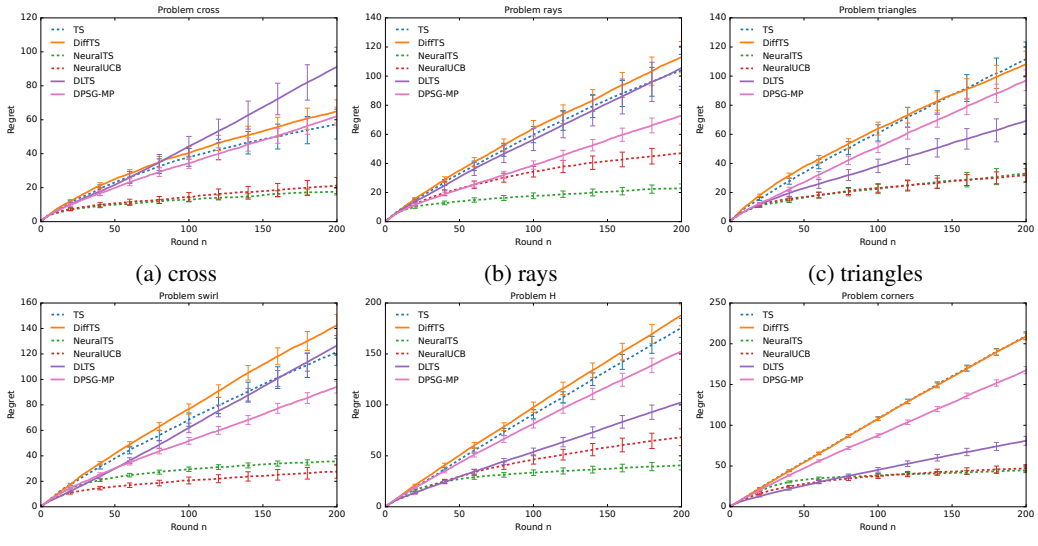
Figure 11: Performance of all algorithms on the cosine nonlinear bandit tasks for the six prior distributions.



(a) cross (b) rays (c) triangles

(d) swirl (e) H (f) corners

Figure 12: Visualization of the learned diffusion prior versus the true prior (under quadratic reward model). The true prior is the MLP parameters learned from the previous task. Samples from our trained model (grey) are overlaid on the ground-truth samples (blue).



(a) cross (b) rays (c) triangles

(d) swirl (e) H (f) corners

Figure 13: Performance of all algorithms on the quadratic nonlinear bandit tasks for the six prior distributions.

H.3 Results for Real-world Application

To validate our approach in more realistic scenarios, we now evaluate our algorithms on the bandit tasks derived from the MovieLens. While prior works such as DiffTS (Kveton et al., 2024) and dTS (Aouali, 2024) also utilize MovieLens for experiments, they primarily focus on synthetic reward functions that are linear or logistic rather than real-world user ratings. Specifically, these methods use the MovieLens-1M dataset (6,040 users and 3,952 movies) and perform low-rank factorization on the sparse rating matrix to obtain 5-dimensional representations for users U and movies V.

In DiffTS (Kveton et al., 2024), to construct the contextual bandit, a movie embedding V_i is selected as the task, and in each round, actions are randomly sampled from user embeddings U. In dTS (Aouali, 2024), the setup is slightly different: a user is randomly selected to define the bandit, and in each round, the algorithm is presented with 100 candidate movie actions. Both methods define their reward functions using either linear UV or logistic $\sigma(UV)$ forms. Consequently, the reward function is explicitly designed rather than derived from real rating labels. In contrast, our setting targets a more complex and realistic environment that requires constructing nonlinear rewards directly from real user data.

In MovieLens, we formulate a contextual bandit task using the MovieLens-25M dataset (1.6×10^5 users and 6×10^4 movies). We construct the bandit problem following previous work (Ban et al., 2021). We first construct a dense rating matrix by selecting the top 2,000 users and top 10,000 movies, and then apply SVD to extract 10-dimensional feature vectors for each user and movie. We treat each user as a distinct task within the multi-task bandit framework. The objective is to identify specific user preferences (specifically, movies with low ratings). In each task, we select a user embedding. In each round, the agent is presented with $K = 10$ arms: 1 'target' arm (a movie with a rating < 2, yielding reward $r = 1$) and 9 'non-target' arms (ratings > 4, yielding reward $r = 0$). To construct the prior, we train task-specific MLPs on the offline binary preference labels where low-rated movies are mapped to reward 1 and high-rated movies to reward 0. A key challenge in diffusing over MLP weights is permutation invariance. To address this and ensure the parameter distribution remains single-modal and learnable, we first train a single 'anchor' MLP and then for all subsequent tasks in the prior dataset, we initialize the MLP weights from this anchor before fine-tuning on the specific task data. This ensures the parameters remain aligned in the weight space. We conduct the experiments for our proposed methods compared with NeuralTS and NeuralUCB over a horizon of 200 rounds with 5 runs. We summarize the results in the Table 5. The experiments are conducted on the RTX A5000 with 24GB memory.

Table 5: Cumulative regret comparison on the MovieLens contextual bandit task over $T = 200$ rounds averaged over 5 runs.

Algorithm	**Regret**
DiffTS	178.20 ± 0.77
NeuralTS	51.20 ± 5.29
NeuralUCB	39.00 ± 10.00
DLTS(Ours)	87.80 ± 11.28
DPSG-MP(Ours)	$\mathbf{28.40 \pm 9.41}$

Results Analysis The results in Table 5 emphasize the challenge of modeling complex, non-linear user preferences compared to synthetic settings. DPSG-MP achieves the lowest cumulative regret among all methods, with 28.40 ± 9.41, outperforming DiffTS (178.20 ± 0.77), NeuralTS (51.20 ± 5.29), and NeuralUCB (39.00 ± 10.00) in mean regret. This suggests that the learned diffusion prior, when combined with likelihood-based guidance and multi-step projection, can provide an effective inductive bias for fast adaptation posterior sampling in high-dimensional neural parameter spaces. We also observe that DLTS improves over DiffTS (87.80 vs. 178.20), but underperforms NeuralTS, NeuralUCB, and DPSG-MP. One possible explanation is that directly approximating the posterior via Langevin dynamics can be challenging in the complex, high-dimensional parameter space of MLP weights. In contrast, the multi-step projection in DPSG-MP may help stabilize likelihood guidance, making it the more effective practical variant on MovieLens.

H.4 Comparison with Learned-Prior Baselines

In this experiment, we study whether learning diffusion priors from historical tasks provides an advantage over gaussian priors learned from the same historical data. We compare our methods with two learned-prior baselines, MixTS (Hong et al., 2022) and Learned-Gaussian TS. We evaluate all methods in the sigmoid-gated nonlinear contextual bandit setting. For MixTS, we fit a two-component full-covariance Gaussian mixture prior to the 8000 underlying two-dimensional task parameters used to generate the historical tasks. For a new task, MixTS maintains posterior uncertainty over both the mixture component and the task parameter, samples a component and its corresponding parameter, and selects the action that maximizes the sampled linear reward. For Learned-Gaussian TS, we instead fit a single full-covariance Gaussian prior to the 8000 historical MLP parameter vectors used in our nonlinear bandit experiments. Given this learned Gaussian prior, we use LMC for posterior sampling under the nonlinear reward model. This baseline provides a more direct comparison for examining whether a simpler learned prior over the same MLP parameter space can achieve similar performance to our diffusion-based approaches.

As shown in Table 6, MixTS has higher mean regret than both DLTS and DPSG-MP on all 6 problems which is expected because of its misspecified linear reward model. The Learned-Gaussian TS baseline obtains lower mean regret than MixTS on 5 of 6 problems, but remains worse than worse than both proposed methods. These results support the effectiveness of our methods leveraging diffusion model compared to the learned-gaussian baselines.

Table 6: Comparison with learned-prior baselines in the sigmoid-gated nonlinear contextual bandit setting. We report cumulative regret at $T = 200$ over 8 independent runs with 64 test tasks.

Problem	DLTS	DPSG-MP	MixTS	Learned-Gaussian TS
Cross	44.33 ± 3.07	$\mathbf{34.87 \pm 0.44}$	87.05 ± 7.23	84.97 ± 18.12
Rays	51.20 ± 2.90	$\mathbf{36.31 \pm 1.09}$	139.00 ± 13.57	124.21 ± 15.31
Triangles	$\mathbf{47.29 \pm 1.79}$	52.20 ± 3.54	135.88 ± 15.76	112.17 ± 32.30
Swirl	53.95 ± 1.74	$\mathbf{38.34 \pm 1.63}$	117.59 ± 8.48	87.45 ± 29.77
H	121.31 ± 5.98	$\mathbf{52.49 \pm 3.43}$	229.40 ± 11.24	116.19 ± 22.79
Corners	57.92 ± 2.35	$\mathbf{38.12 \pm 1.27}$	263.05 ± 27.09	316.87 ± 54.27

H.5 High-Dimensional Neural Reward Models.

The synthetic nonlinear experiments above are generated from a two-dimensional task parameter, so the resulting MLP parameters may still reflect low-dimensional structure induced by this construction. To test our methods without this restriction, we consider an additional setting in which each task is directly represented by an MLP with two hidden layers of width 32 and 1185 parameters. Instead of generating the reward function from an underlying two-dimensional parameter, we directly sample the standardized MLP parameter vector from an equally weighted four-component Gaussian mixture, $z \sim \frac{1}{4}\sum_{c=1}^{4}\mathcal{N}(3Qv_c, I_{1185})$, where $v_c \in \{(1,1,1),(1,-1,-1),(-1,1,-1),(-1,-1,1)\}/\sqrt{3}$ are four unit vectors corresponding to the vertices of a regular tetrahedron, and $Q \in \mathbb{R}^{1185\times 3}$ has orthonormal columns. Each Gaussian component has covariance I_{1185} and therefore has nonzero variance in all 1185 parameter-space directions, rather than being restricted to the two-dimensional task construction used in the preceding experiments.

As shown in Table 7, DLTS and DPSG-MP achieve lower mean cumulative regret than NeuralTS and NeuralUCB at both $T = 50$ and $T = 200$. These results demonstrate that the effectiveness of our methods is not limited to the low-dimensional tasks, but also extends to directly sampled high-dimensional neural reward models.

H.6 Sensitivity to Diffusion Steps L

We first investigate the effect of the number of diffusion steps L while keeping other hyperparameters fixed. These experiments are performed on the sigmoid-gated nonlinear bandit benchmark, a challenging setting from Section 5.1. We report the cumulative regret at $T = 200$, averaged over

Table 7: Cumulative regret on directly sampled neural reward models. Each task is defined by an MLP with two hidden layers of width 32 and 1185 parameters is sampled directly from a full-rank Gaussian mixture. Results are mean ± standard deviation over 8 runs with 64 tasks.

Algorithm	$T = 50$	$T = 200$
DLTS	$\mathbf{12.30 \pm 1.04}$	$\mathbf{24.84 \pm 0.71}$
DPSG-MP	13.88 ± 0.48	27.84 ± 1.33
NeuralTS	14.18 ± 1.36	36.66 ± 6.14
NeuralUCB	16.53 ± 1.98	42.10 ± 3.20

64 tasks across 8 independent runs. As shown in Table 8, increasing L consistently improves the performance of both DLTS and DPSG-MP across all three environments. These results suggest that a more refined reverse generation process improves the quality of posterior sampling and allows the likelihood information from the online history to be incorporated more effectively. Based on these results, we use $L = 100$ as the default setting in our main experiments.

Table 8: Results of ablation study on Diffusion Steps L. Reported values are cumulative regret at $T = 200$. The regrets are averaged over 64 tasks across 8 independent runs.

Environment	**Algorithm**	**Diffusion Steps (L)**		
		$L = 20$	$L = 50$	$L = 100$
Rays	DLTS	136.08 ± 3.18	53.51 ± 4.42	$\mathbf{51.20 \pm 2.90}$
	DPSG-MP	63.90 ± 0.72	50.28 ± 1.09	$\mathbf{36.31 \pm 1.09}$
Triangles	DLTS	433.16 ± 7.51	97.68 ± 5.95	$\mathbf{47.29 \pm 1.79}$
	DPSG-MP	86.57 ± 0.66	73.09 ± 1.19	$\mathbf{52.20 \pm 3.54}$
Swirl	DLTS	732.70 ± 7.29	125.55 ± 4.06	$\mathbf{53.95 \pm 1.74}$
	DPSG-MP	75.45 ± 0.59	52.47 ± 0.51	$\mathbf{38.34 \pm 1.63}$

H.7 Sensitivity to Inner-Loop Updates K_ℓ

We then investigate the effect of the number of inner-loop updates K_ℓ which are LMC steps for DLTS and projection steps for DPSG-MP, fixing the Diffusion Steps $L = 100$. The results, summarized in Table 9, indicate that a small number of steps (e.g., $K_\ell \in \{1, 2\}$) is generally sufficient to achieve strong performance. Further increasing K_ℓ yields diminishing returns and may even degrade performance in certain cases (e.g., DLTS on Triangles with $K_\ell = 5$), likely due to overfitting the local likelihood approximation. Consequently, we recommend a small K_ℓ as it offers the best trade-off between regret minimization and computational efficiency.

Table 9: Results of ablation study on Inner-Loop Updates K_ℓ with fixed Diffusion Steps $L = 100$. Reported values are cumulative regret at $T = 200$. The regrets are averaged over 64 tasks across 8 independent runs.

Environment	**Algorithm**	**Inner-Loop Steps (K_ℓ)**			
		$K_\ell = 1$	$K_\ell = 2$	$K_\ell = 5$	$K_\ell = 10$
Rays	DLTS	51.20 ± 2.90	$\mathbf{43.72 \pm 1.58}$	63.24 ± 3.48	55.87 ± 3.04
	DPSG-MP	39.59 ± 0.80	39.47 ± 0.85	37.36 ± 0.33	$\mathbf{36.31 \pm 1.09}$
Triangles	DLTS	79.60 ± 2.49	52.65 ± 5.32	54.86 ± 7.49	$\mathbf{47.29 \pm 1.79}$
	DPSG-MP	54.94 ± 6.35	51.28 ± 2.44	$\mathbf{49.51 \pm 1.96}$	52.20 ± 3.54
Swirl	DLTS	90.20 ± 3.63	78.81 ± 0.77	66.49 ± 2.34	$\mathbf{53.95 \pm 1.74}$
	DPSG-MP	$\mathbf{34.80 \pm 1.43}$	36.46 ± 0.87	34.81 ± 0.93	38.34 ± 1.63

H.8 Long Horizon Evaluation

To validate the long-term robustness of our approach, we extend the evaluation of our most challenging nonlinear experiment, the sigmoid-gated reward model, to a horizon of $T = 2000$ rounds (increased from $T = 200$). The cumulative regret results are summarized in the Table 10 across various prior distributions.

Table 10: Cumulative regret at horizon 2000 on sigmoid-gated nonlinear bandits.

Environment	LinTS	DiffTS	NeuralTS	NeuralUCB	DLTS (Ours)	DPSG-MP (Ours)
Cross	719.30 ± 329.99	899.33 ± 622.08	441.20 ± 264.00	413.51 ± 383.25	$\mathbf{221.01 \pm 130.05}$	252.91 ± 7.50
Rays	1392.70 ± 545.47	1753.93 ± 883.84	501.88 ± 403.60	396.61 ± 304.10	$\mathbf{112.22 \pm 27.26}$	355.43 ± 98.89
Triangles	942.77 ± 505.50	2873.07 ± 530.49	$\mathbf{151.44 \pm 110.73}$	1717.32 ± 532.37	513.40 ± 137.74	189.37 ± 16.48
Swirl	1388.62 ± 723.92	1012.04 ± 231.75	295.03 ± 304.34	997.91 ± 597.07	483.63 ± 234.47	$\mathbf{176.44 \pm 33.48}$
H	1702.86 ± 1043.99	3120.56 ± 827.30	$\mathbf{114.73 \pm 25.09}$	1303.26 ± 603.24	336.62 ± 17.21	323.96 ± 36.16
Corners	1539.22 ± 635.22	2057.89 ± 1181.21	$\mathbf{244.09 \pm 105.71}$	3515.52 ± 953.39	247.07 ± 92.16	258.20 ± 24.41

The extended horizon experiments confirm the robustness of our framework. In the 'Cross' and 'Rays' environments, both our methods (DLTS and DPSG-MP) clearly outperform all baselines, achieving the lowest cumulative regret. In 'Swirl' environment, DPSG-MP clearly outperforms all baselines. In the remaining three scenarios, our approach consistently achieves top-two performance. Specifically, in 'Corners', NeuralTS yields regret comparable to DPSG-MP (244.09 vs. 258.20) but suffers from significantly higher instability (± 105.71 vs. ± 24.41). In general, the neural baselines exhibit much higher variance across tasks, whereas our diffusion-based methods provide a more stable and reliable exploration strategy over long horizons.

H.9 Sensitivity to Amount of Historical Data

We further evaluate the sensitivity of our methods to the amount of historical data used to train the diffusion prior. We train three separate diffusion priors using 10%, 50%, and 100% of the historical training data, while keeping the model architecture and other training settings unchanged.

We evaluate the resulting priors with DLTS and DPSG-MP on the sigmoid-gated nonlinear contextual bandit under the Swirl and Triangles prior distributions. We follow the same evaluation protocol as in the main experiments, using a horizon of $T = 200$ and evaluating over 64 test tasks across 8 independent runs. Table 11 reports the mean cumulative regret with standard deviation.

Table 11: Sensitivity to the amount of historical data used to train the diffusion prior. We report cumulative regret at $T = 200$ in the sigmoid-gated nonlinear contextual bandit setting.

Environment	Method	10%	50%	100%
Swirl	DLTS	60.45 ± 0.33	60.10 ± 0.51	$\mathbf{53.95 \pm 1.74}$
	DPSG-MP	41.54 ± 0.30	38.38 ± 0.35	$\mathbf{38.34 \pm 1.63}$
Triangles	DLTS	47.73 ± 0.77	47.49 ± 0.61	$\mathbf{47.29 \pm 1.79}$
	DPSG-MP	57.66 ± 0.69	$\mathbf{50.97 \pm 0.95}$	52.20 ± 3.54

The results show that using less historical data can hurt performance, but the effect varies across settings. DLTS benefits from more data on Swirl but changes little on Triangles, while DPSG-MP performs similarly with 50% and 100% of the data and degrades more with only 10%.