CLEAN-SAMPLE MARKOV CHAIN FOR REWARD-GUIDED DISCRETE DIFFUSION SAMPLING

Anonymous authors
Paper under double-blind review

ABSTRACT

Discrete diffusion models have recently emerged as a powerful class of generative models for chemistry and biology data. In these fields, the goal is to generate various samples with high rewards (e.g., binding affinity between peptides and proteins), making reward-based guidance crucial. Most existing methods are based on guiding the diffusion model using intermediate rewards but tend to underperform since intermediate rewards are noisy due to the non-smooth nature of reward functions used in scientific domains. To address this, we propose **Clean-Sample Markov Chain** (CSMC) Sampler, a method that performs effective test-time reward-guided sampling for discrete diffusion models, enabling local search without relying on intermediate rewards. CSMC constructs a Markov chain of clean samples using the Metropolis-Hastings algorithm such that its stationary distribution is the reward-weighted distribution. We design a proposal distribution by sequentially applying the forward and backward diffusion processes, making the acceptance probability tractable. Experiments on peptide and biological sequence generation with various reward functions demonstrate that our method consistently outperforms prior approaches that rely on intermediate rewards.

1 INTRODUCTION

Discrete diffusion models have recently emerged as a powerful generative framework for discrete data, showing particular promise in chemistry and biology for generating complex structures such as molecules and DNA sequences. Unlike autoregressive models that assume a canonical left-to-right ordering on the data, discrete diffusion models are more naturally suited for scientific domains whose data (e.g., molecules or DNA sequences) lack a natural fixed ordering. For instance, the widely used SMILES (Weininger, 1988) representation for molecules is based on heuristic rules such as depth-first search which does not use a unified fixed ordering (Lee et al., 2025).

In many applications in chemistry and biology, there are well-defined notions of *quality* for the data; for example, binding affinity for molecule binders (small molecules, peptides, proteins) or enhancer activity (Taskiran et al., 2024; Wang et al., 2025) for DNA sequences. Thus, generative models must not only produce *natural* samples that resemble the training data but also achieve high-quality scores according to such domain-specific criteria.

In the emerging regime of test-time scaling, quality considerations are incorporated by defining a reward function and optimizing it during inference through reward-guided sampling. The simplest strategy is Best-of-N sampling, where N samples are generated, and the one with the highest reward is selected. However, this brute-force method is inefficient since it does not perform any structured guidance. Recent works (Kim et al., 2025b; Wu et al., 2023; Yu et al., 2023; Bansal et al., 2024; Chung et al., 2023) have instead proposed guidance using *process rewards* or *intermediate rewards*, which are computable at intermediate steps of generation.

While methods involving intermediate rewards are technically applicable to discrete diffusion models, they pose particular challenges for many types of discrete chemistry and biology data as reward functions in these domains are often non-smooth, meaning that small perturbations in the data can cause large changes in the reward. For example, in molecular structures, modifying even a single element in the string representation can render the entire molecule invalid, collapsing the reward to zero[1], as shown in Fig. 1 (left). As a result, relying on intermediate rewards does not provide an effective local search strategy in these cases.

[1]For negative rewards, we define the invalid reward to be a large negative number, such as -100.

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.

The key question that arises here is how to enable local search in reward-guided generation without relying on intermediate rewards. We introduce **Clean-Sample Markov Chain (CSMC) Sampler** which performs iterative search over clean data samples using the Metropolis–Hastings (MH) algorithm. To propose a new clean sample from an existing one, we use forward–reverse diffusion which applies the forward process to corrupt clean data followed by running the reverse process to obtain a new sample. While the acceptance probability for the MH algorithm is intractable due to intractable clean sample probabilities, we show that, assuming a well-trained diffusion model, the forward-reverse proposal makes the acceptance probability tractable, enabling efficient sampling.

We validate CSMC on therapeutic peptide and biological sequence generation across multiple reward functions. CSMC achieves the highest reward in all settings across matched wall-clock time, forming the Pareto front. Furthermore, we demonstrate that CSMC is compatible with fine-tuned diffusion models, providing an axis of scaling complementary to training-based alignment.

2 Background

2.1 Discrete Diffusion Models

Diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021) learn to reverse a forward Markov process. Although originally designed for continuous spaces, diffusion models have also been successfully applied to discrete state spaces (Austin et al., 2021; Campbell et al., 2022; Lou et al., 2024; Sahoo et al., 2024; Schiff et al., 2025; Shi et al., 2024).

These discrete diffusion models are characterized by forward transition matrices $\boldsymbol{Q}_t$. Let $\overline{\boldsymbol{Q}}_t = \boldsymbol{Q}_1\boldsymbol{Q}_2\cdots\boldsymbol{Q}_t$. These transition matrices define the forward process:

$$p_{t|t-1}(x_t|x_{t-1}) = \text{Cat}\left(x_t; p = x_{t-1}\boldsymbol{Q}_t\right),$$
$$p_t(x_t|x_0) = \text{Cat}\left(x_t; p = x_0\overline{\boldsymbol{Q}}_t\right),$$

where $\text{Cat}(\cdot; p)$ denotes the categorical distribution with probabilities given by p.

A neural network $p_\theta(x_0|x_t) \approx p(x_0|x_t)$ is learned to invert the forward process, and denoising is performed by the following parameterization:

$$p_\theta(x_{t-1}|x_t) \propto \sum_{x_0} p(x_{t-1}, x_t|x_0)p_\theta(x_0|x_t).$$

Notably, the learned neural network $p_\theta(x_0|x_t)$ predicts a distribution over x_0 given x_t which enables sampling x_0-predictions $\hat{x}_0(x_t) \sim p_\theta(x_0|x_t)$.

Masked Diffusion Models (MDMs) (Sahoo et al., 2024). MDMs define the forward process by progressively replacing tokens with a special mask token. The denoising model learns to recover the original sequence from these masked inputs.

2.2 Reward-guided diffusion samplers

The simplest method is Best-of-N (BoN) sampling (Stiennon et al., 2020), which generates N samples independently and selects the one with the highest reward. While BoN is often highly effective, it does not guide the denoising trajectory using the reward, resulting in inefficient search, especially when high-reward samples lie in low-density regions unlikely to be sampled by the model.

On the other hand, particle-based methods (Ma et al., 2025; Kim et al., 2025a; Singhal et al., 2025; Li et al., 2025), such as SMC (Doucet et al., 2001; Naesseth et al., 2019; Del Moral, 2004) and SVDD (Li et al., 2025), incorporate reward signals *during* the denoising process through a reward-based resampling step. These methods propose N particles (samples) at each step and resamples them with probability based on their expected reward, encouraging local exploration around potentially promising samples. Ou et al. (2026) extends this approach by leveraging reward gradients to improve the proposal. Similarly, beam search (Oshima et al., 2025) applies classical search algorithms with the expected reward as the scoring criterion. A key challenge with these methods is that they require computing **intermediate rewards** on noisy samples rather than on fully denoised samples. To address this, these approaches use the diffusion model's $p_\theta(x_0|x_t)$ prediction to draw approximate clean

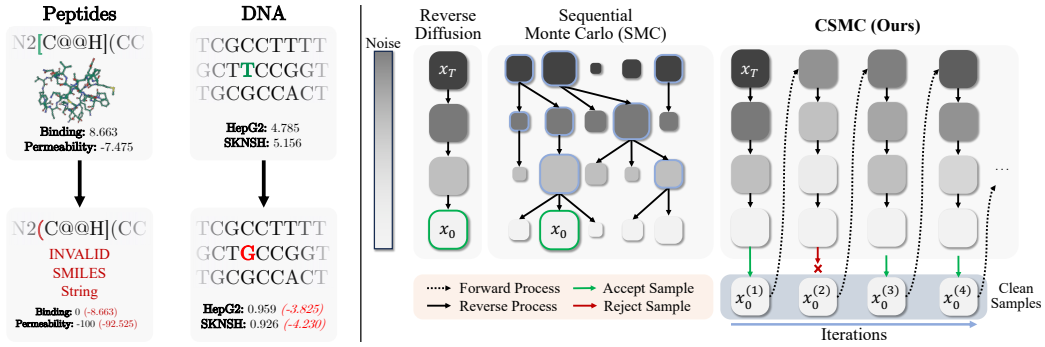

Figure 1: **Left:** In scientific applications, the rewards defined on discrete spaces are highly sensitive to small perturbations. A one or two-character change to a SMILES string or DNA sequence can result in drastically different rewards. **Right:** Reverse diffusion and typical inference-time scaling methods such as SMC (Kim et al., 2025b) guide samples through noise levels by constructing a Markov chain beginning with pure noise and ending with clean samples. Our CSMC constructs a Markov chain of clean samples by successively applying the forward and reverse processes sequentially at each step, bypassing the need for intermediate rewards and evaluating the reward directly on clean samples while leveraging information from past samples for guidance.

samples to compute the expected reward. This enables reward-guidance *during* the denoising process, but is bottlenecked by the accuracy of the intermediate rewards, which is inaccurate in many scientific applications. As a hybrid approach, RERD (Uehara et al., 2025) iteratively runs SVDD on a global particle pool, resamples the resulting clean samples based on its clean reward, and renoises them to a fixed time t for the next iteration. Although its global resampling uses clean rewards, each SVDD pass still relies on inaccurate intermediate rewards, affecting its performance.

In this work, we completely bypass the need for intermediate rewards by using the Metropolis-Hastings algorithm (Metropolis et al., 1953; Hastings, 1970; Robert & Casella, 2009) to construct a Markov chain of *clean* samples that converges to the desired target distribution. Our method only requires the computation of *clean rewards* on clean samples, which can then be used to efficiently guide the Markov chain. While DTS (Jain et al., 2025) also uses clean reward by applying MCTS (Kocsis & Szepesvári, 2006) with full diffusion rollout for value backpropagation, sampling is computationally expensive due to search tree construction.

Some recent works also apply MCMC techniques to diffusion reward guidance. Chu et al. (2025) applied split Gibbs sampling (Vono et al., 2019) to discrete diffusion models. However, their method is only applicable to uniform discrete diffusion models. Wang et al. (2026) used MCMC for initial noise optimization in Gaussian flow models. However, their method is designed for Gaussian flow model and cannot be directly applied to discrete diffusion models. In this work, we design an alternate diffusion MCMC algorithm which can be applied to masked discrete diffusion models.

3 Clean-Sample Markov Chain Sampler

Let the reward function be denoted by $r(\cdot) : \mathcal{X} \to \mathbb{R}$ where $\mathcal{X}$ is the domain of the data, and $\Delta(\mathcal{X})$ denote the set of all probability distributions defined on $\mathcal{X}$. Given a reward function $r(x)$, our objective is to generate natural samples with high rewards leveraging a pretrained generative model $p^{\text{pre}}(\cdot)$. To accomplish this, we sample from the reward-weighted distribution $p_\beta(x)$:

$$p_\beta(x) := \arg\max_{p \in \Delta(\mathcal{X})} \mathbb{E}_{x\sim p(\cdot)}\left[r(x)\right] - \beta D_{\text{KL}}(p(\cdot)\|p^{\text{pre}}(\cdot)) \\ \propto \exp(r(x)/\beta)p^{\text{pre}}(x),$$

where $\beta > 0$ is the temperature hyperparameter controlling the "naturalness" of the samples through KL-regularization with the pretrained model.

Previous methods, including particle-based methods, perform guidance by utilizing intermediate rewards $r(\hat{x}_0(x_t))$ computed from the x_0-prediction $\hat{x}_0(x_t)$ at x_t as an approximation to the clean sample. However, intermediate rewards are often inaccurate and noisy in many scientific applications

due to the non-smooth reward functions. In scientific applications, even a slight perturbation can significantly impact the reward. For instance, when generating peptides based on the SMILES (Weininger, 1988) representation, modifying a single token on a high-reward peptide can result in an invalid peptide with zero reward, or modifying a single token in a high-reward DNA sequence can result in a low-reward sequence, as shown in Fig. 1 (left). This is in stark contrast to rewards defined on continuous space where a slight perturbation smoothly affects the reward (e.g., small perturbations to some pixel values of an image leave the semantics of the image unchanged). Due to the sensitivity of the reward, the x_0 prediction must be accurate, otherwise the intermediate rewards will be uninformative. A sample whose intermediate reward is zero may be prematurely removed by a particle-based method even though a single-token change may yield a high-reward sample (Fig. 1, left).

In these cases, it is more beneficial to leverage only clean reward $r(x_0)$ computed on a clean sample x_0 rather than on an approximation $\hat{x}_0(x_t)$. One such method leveraging clean rewards is Best-of-N (BoN) sampling (Stiennon et al., 2020). However, BoN does not provide guidance during the denoising process, resulting in inefficient exploration. Its effectiveness is therefore limited to what the pretrained model can already generate: if high-reward samples lie in low-density regions of the model's distribution, they are unlikely to ever be produced, even when many samples are drawn. A natural question arises: **How can we leverage clean rewards while using information from past samples for guidance?**

Our proposed method, **C**lean-**S**ample **M**arkov **C**hain **(CSMC)** Sampler, answers this question by using the Metropolis-Hastings (MH) algorithm to construct a Markov chain of clean samples that converges to the reward-weighted distribution $p_\beta(x_0)$. By using only clean samples, CSMC leverages accurate clean rewards. By using the MH algorithm to iteratively refine the samples, the clean rewards of past samples can be used for guidance.

3.1 Bypassing the Intermediate Rewards

Previous methods require intermediate rewards because the denoising trajectory of diffusion models output clean samples only at the very end. A natural solution is to instead explore the clean data space by constructing a Markov chain of *clean* samples $\{x_0^{(i)}\}_i$ that converges to the target distribution $p_\beta(x_0)$. At each iteration, rewards can then be computed directly on clean samples, yielding clean rewards $r(x_0)$ used to guide the chain towards the target distribution. To accomplish this, we propose to use the Metropolis-Hastings (MH) algorithm.

The Metropolis-Hastings (MH) Algorithm. Suppose we want to sample from the unnormalized target distribution $p_\beta(x_0)$. The Metropolis-Hastings (MH) algorithm (Metropolis et al., 1953; Hastings, 1970; Robert & Casella, 2009) is a classical and widely-used MCMC technique for sampling from unnormalized probability distributions by constructing a Markov chain that converges to the target distribution. Two ingredients are necessary for the MH algorithm: (1) a proposal distribution and (2) an acceptance probability. The proposal distribution $q(x_0'|x_0)$ is used to generate the next candidate $x_0' \sim q(x_0'|x_0)$ given the current sample x_0. After generating the proposal candidate, we decide whether or not to accept this proposal with probability $A(x_0'|x_0)$ defined as

$$A(x_0'|x_0) = \min(1, \alpha), \quad \alpha = \frac{p_\beta(x_0')q(x_0|x_0')}{p_\beta(x_0)q(x_0'|x_0)}.$$

If x_0' is accepted, we update the current state of the chain to x_0'. Otherwise, we retain x_0. By repeating these two steps, the MH algorithm constructs a Markov chain whose stationary distribution is the target distribution (Murphy, 2023). For more background and proof of convergence on the MH algorithm, please refer to App. A.2. In order to efficiently use the MH algorithm, we require a proposal distribution that is easy to sample from as well as an acceptance probability that can be efficiently computed.

3.2 **Proposal Distribution**: Forward-Reverse Diffusion

We require a proposal distribution which lets us sample a new clean sample x_0' from a given clean sample x_0. Let $p^{\text{pre}}(x_0)$ and $p_t^{\text{pre}}(x_t)$ denote the pretrained probability distribution at time 0 and t, respectively. Let $\overrightarrow{p}_t(x_t|x_0)$ denote the probability of x_t given x_0 under the forward diffusion process, and $\overleftarrow{p}(x_0|x_t)$ denote the probability of x_0 given x_t under the reverse diffusion process.

We construct the proposal as follows. Starting from the current clean sample x_0, we choose a random time $t \sim \mathcal{U}(t_{\text{lo}}, t_{\text{hi}})$ (where t_{lo} and t_{hi} are user-defined parameters) and apply the forward process to

obtain a noisy auxiliary sample $x_t \sim \overrightarrow{p}_t(\cdot|x_0)$. Then, we run reverse diffusion to obtain a new clean sample $x_0' \sim \overleftarrow{p}(\cdot|x_t)$. This proposal distribution can be written as

$$q_{x_t}(x_0'|x_0) := \overrightarrow{p}_t(x_t|x_0)\overleftarrow{p}(x_0'|x_t), \tag{1}$$

where $t \sim \mathcal{U}(t_{\text{lo}}, t_{\text{hi}})$ and $x_t \sim \overrightarrow{p}_t(\cdot|x_0)$ are resampled at each step. This proposal distribution can be interpreted as exploring locally around x_0, with smaller t encouraging local exploitation around the current sample, while larger t encourages global exploration.

Remark 1. Although we resample t and x_t at each step, our Markov chain is *not* time-inhomogeneous: it is equivalent to the time-homogeneous Markov chain defined by the time-invariant proposal $q(x_0'|x_0) = \mathbb{E}_{t\sim\mathcal{U}(t_{\text{lo}},t_{\text{hi}})}\mathbb{E}_{x_t\sim\overrightarrow{p}_t(\cdot|x_0)}[\overleftarrow{p}(x_0'|x_t)]$.

3.3 Acceptance Probability

With the forward-reverse diffusion as our proposal function, we can rewrite the log acceptance ratio as

$$\log \alpha(x_0'|x_0) = \frac{r(x_0') - r(x_0)}{\beta} + \underbrace{\log\left(\frac{p^{\text{pre}}(x_0')\overrightarrow{p}(x_t|x_0')\overleftarrow{p}(x_0|x_t)}{p^{\text{pre}}(x_0)\overrightarrow{p}(x_t|x_0)\overleftarrow{p}(x_0'|x_t)}\right)}_{=:\chi_{x_t}(x_0'|x_0)}. \tag{2}$$

The first term can be efficiently computed since we have access to the reward function. However, the second term $\chi_{x_t}(x_0'|x_0)$ is not tractable since $p^{\text{pre}}(x_0)$ and $p^{\text{pre}}(x_0')$ are intractable. One way to compute the acceptance probability is to model $\chi_{x_t}(x_0'|x_0)$ by training a discriminator as follows. Define auxiliary positive and negative distributions

$$p^+(x_0, x_0', x_t, t) = p^{\text{pre}}(x_0')\overrightarrow{p}(x_t|x_0')\overleftarrow{p}(x_0|x_t),$$
$$p^-(x_0, x_0', x_t, t) = p^+(x_0', x_0, x_t, t).$$

$p^+(x_0, x_0', x_t, t)$ gives the probability of sampling x_0' and obtaining x_0 from the forward-reverse diffusion with forward noising time t and intermediate noisy state x_t. $p^-(x_0, x_0', x_t, t)$ is the reverse: the probability of sampling x_0 and obtaining x_0' from x_0 by applying forward-reverse diffusion.

We collect the training data as follows. Positive samples are obtained by sampling x_0' using the pretrained diffusion model, noising it to a randomly sampled time t to obtain x_t, and running reverse diffusion to obtain the clean sample x_0. The negative samples are obtained by swapping x_0 and x_0' in the positive samples. The discriminator is a binary classifier trained to classify the positive and negative samples:

$$D^\star = \arg\max_D \mathbb{E}_{(x_0,x_0',x_t,t)\sim p^+}\Big[\log D(x_0, x_0', x_t, t) + \log(1 - D(x_0', x_0, x_t, t))\Big] \tag{3}$$

$\chi_{x_t}(x_0'|x_0)$ can be obtained from the discriminator as follows:

Theorem 3.1. *Let $D^\star : \mathcal{X}^3 \times \mathbb{R} \to [0, 1]$ be the optimal discriminator in Eq. 3. Then,*

$$\chi_{x_t}(x_0'|x_0) = \log\left(\frac{D^\star(x_0, x_0', x_t, t)}{1 - D^\star(x_0, x_0', x_t, t)}\right) \tag{4}$$

For the proof of Theorem 3.1 and more training details of the discriminator, please refer to App. B.2. While the discriminator can model the intractable $\chi_{x_t}(x_0'|x_0)$, it requires additional training of the discriminator for each pretrained model. We propose an alternative *training-free* approach below.

As shown in App. B.1, a well-trained diffusion model matches the forward and reverse process marginals: $\overleftarrow{p}(x_0|x_t) = \overrightarrow{p}(x_0|x_t)$. Under this matching forward-reverse marginals assumption, we show that $\chi_{x_t}(x_0'|x_0)$ simplifies to zero:

$$\begin{aligned}
\chi_{x_t}(x_0'|x_0) &= \log\left(\frac{p^{\text{pre}}(x_0')\overrightarrow{p}(x_t|x_0')\overleftarrow{p}(x_0|x_t)}{p^{\text{pre}}(x_0)\overrightarrow{p}(x_t|x_0)\overleftarrow{p}(x_0'|x_t)}\right) \\
&= \log\left(\frac{p_t^{\text{pre}}(x_t)\overleftarrow{p}(x_0'|x_t)\overleftarrow{p}(x_0|x_t)}{p_t^{\text{pre}}(x_t)\overleftarrow{p}(x_0|x_t)\overleftarrow{p}(x_0'|x_t)}\right) \\
&= 0,
\end{aligned}$$

The second line follows from our assumption $\overleftarrow{p}(x_0|x_t) = \overrightarrow{p}(x_0|x_t)$, which ensures $p^{\text{pre}}(x_0)\overrightarrow{p}(x_t|x_0) = p_t^{\text{pre}}(x_t)\overleftarrow{p}(x_0|x_t)$, as explained in more detail in App. B.1.

Thus, the acceptance probability simplifies to

$$A(x_0'|x_0) = \min(1, \alpha), \quad \alpha = \exp\left(\frac{r(x_0') - r(x_0)}{\beta}\right), \tag{5}$$

which can now be efficiently computed in a training-free manner.

Fig. 4f and App. B.2 show that using the acceptance probability obtained via the trained discriminator does not significantly impact the performance of CSMC. This shows that, in practice, our simplified acceptance probability is sufficient for reward-guided sampling.

3.4 Implementation

CSMC begins by running full reverse diffusion to obtain a clean initial sample $x_0^{(1)}$. At each step, we generate a candidate clean sample using our forward-backward diffusion process and choose to accept or reject the candidate based on the simplified MH acceptance probability given in Eq. 5. In practice, we implement the reverse diffusion using M steps where M is a user-defined hyperparameter. This gives CSMC three user-defined variables: the time window $[t_{\text{lo}}, t_{\text{hi}}]$ used for forward noising, the number of reverse steps M, and the temperature β. Running this algorithm results in a chain of clean samples whose stationary distribution is the reward-weighted distribution $p_\beta(x_0)$. CSMC is visualized in Fig. 1 (Right), and the full algorithm is shown in Alg. 1.

Parallel Tempering. While decreasing the temperature β improves the performance by sharpening p_β around high-reward modes, using small $\beta \ll 1$ can trap CSMC in a single mode, resulting in insufficient exploration and slow mixing of the Markov chain. A standard fix is to use annealing methods that incorporate multiple temperatures during sampling. One such method is parallel tempering (Swendsen & Wang, 1986), which maintains L chains at different temperatures $\beta_1, \ldots, \beta_L$ where $\beta_1 < \cdots < \beta_L$, and β_1 is the temperature corresponding to our target distribution. We denote the samples of chain ℓ by $x_{0,\ell}$. Each chain is independently updated using the CSMC transition. After each step, a swap is proposed to switch samples on different chains ℓ and ℓ', with acceptance probability

$$A_{\text{swap}}(x_{0,\ell}, x_{0,\ell'}) = \min\left(1, \exp\left((r(x_{0,\ell'}) - r(x_{0,\ell}))\left(\frac{1}{\beta_\ell} - \frac{1}{\beta_{\ell'}}\right)\right)\right).$$

These swaps can be performed for multiple chains in parallel as well. For more details, please refer to App. A.3. High-temperature chains encourage exploration around various modes while the swaps allow the chains to communicate and the low-temperature chains to escape its current mode. Additionally, parallel tempering enables parallelization of CSMC, improving performance, as shown in Fig. 4d. In our experiments, we run Alg. 1 with parallel tempering. The algorithm for CSMC with parallel tempering is shown in Alg. 3 in App. A.3.

4 Experiments

We conduct experiments on therapeutic peptide generation with PepTune (Tang et al., 2025), and biological sequence design using an MDM trained on the MPRA dataset (Gosai et al., 2024). For each method, we report both its given reward (reward used during guidance) as well as its heldout reward (reward not seen during guidance). We compare the following training-free reward-guided sampling methods:

- **Pretrained models:** Samples are generated using the pretrained model.
- **Best-of-N (BoN) (Stiennon et al., 2020):** N samples are generated and the one with the highest reward is selected.
- **SMC (Doucet et al., 2001):** A set of N particles are updated using diffusion reverse sampling and periodically resampled based on the intermediate reward.
- **SMC+Grad (Ou et al., 2026):** For differentiable rewards, SMC with a gradient-augmented proposal $q(x_{t-1}|x_t) \propto p^{\text{pre}}(x_{t-1}|x_t)\exp(x_{t-1}^\top \nabla_x r(x_t))$.
- **SVDD (Li et al., 2025):** N candidate samples are generated at each step and a sample is selected by randomly choosing a sample with probability based on its intermediate reward.

Algorithm 1 CSMC (Clean-Sample Markov Chain Sampler)

$x_0^{(1)} \sim p(x_T) \prod_{t=1}^{T} p(x_{t-1}|x_t)$
for $k = 1, \ldots, K-1$ **do**
  $t_0 \sim \mathcal{U}(t_{\text{lo}}, t_{\text{hi}})$
  $x_{t_0}^{(k)} \sim p_{t_0}(\cdot|x_0^{(k)})$
  $\tilde{x}_0^{(k+1)} \sim \prod_{i=1}^{M} p(x_{t_{i-1}}^{(k)}|x_{t_i}^{(k)})$
  $\alpha \leftarrow \exp((r(\tilde{x}_0^{(k+1)}) - r(x_0^{(k)}))/\beta)$
  $A \leftarrow \min(1, \alpha)$
  $\rho \sim \mathcal{U}(0, 1)$
  if $\rho < A$ **then**
    $x_0^{(k+1)} \leftarrow \tilde{x}_0^{(k+1)}$
  else
    $x_0^{(k+1)} \leftarrow x_0^{(k)}$
  end if
end for
return $x_0^{(K)}$

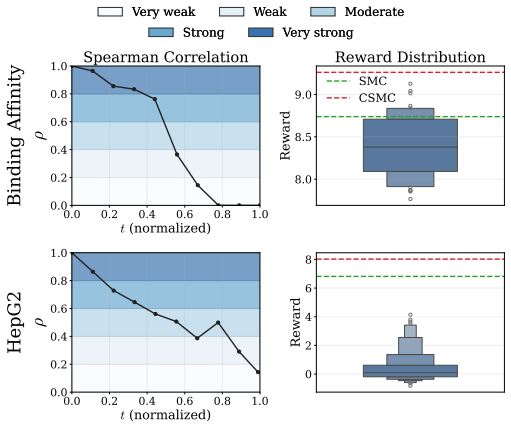

Figure 2: PepTune binding affinity and MPRA HepG2 intermediate reward Spearman correlation (left) and pretrained model reward distribution (right).

- **RERD (Uehara et al., 2025):** Multiple particles are maintained. Each particle is updated via SVDD, global resampling across clean samples is performed, and noise is added to obtain noisy samples. This process is iteratively repeated to refine the samples.
- **DTS (Jain et al., 2025):** Diffusion reward guidance through search based on Monte Carlo tree search (Kocsis & Szepesvári, 2006).
- **Beam Search (Oshima et al., 2025):** Tree search which proposes multiple candidates at each level through diffusion, and retains the top B samples based on intermediate reward value.
- **CSMC (Ours):** Alg. 1 with parallel tempering which constructs a Markov chain targeting $p_\beta(x_0)$.

4.1 Settings

Peptide Generation. Peptides are short chains of amino acids used to orchestrate hormone activity, immune defense, tissue repair, and many other bodily functions. We use PepTune (Tang et al., 2025), an MDM for generating therapeutic peptide SMILES, as our base model. We evaluate on two rewards: binding affinity and permeability. Binding affinity measures the strength of the binding interaction between the generated peptide and the target protein **AMHR2**, and permeability measures the membrane permeability of the peptide. For the given reward used for guidance, we use the therapeutic property predictor proposed by Tang et al. (2025). For the heldout reward, we use the PeptiVerse (Zhang et al., 2026) binding affinity and permeability predictor. For peptide generation, some samples may be invalid SMILES strings. We report only the mean reward of valid SMILES, and report the validity of each method in App. D.1.

Biological Sequence Generation. We pretrain the discrete diffusion models on the DNA dataset provided by Gosai et al. (2024) (which we refer to as MPRA) which measures the enhancer activity of ∼700,000 DNA sequences using massively parallel reporter assays (MPRA). An Enformer model (Avsec et al., 2021) is trained to predict the enhancer activity level in the HepG2 and SK-N-SH cell lines which are used as reward functions. We use the model trained on two different subsets of data, using one model for reward guidance and the other as a heldout reward. Higher predicted activity indicates better performance.

4.2 Results

Fig. 3 compares reward against wall-clock time to show the scaling properties of each reward-guidance method. Across all rewards, CSMC consistently outperforms all baselines across matched wall-clock times, forming the Pareto front for both given and heldout rewards.

SMC, SVDD, RERD, and beam search all rely on intermediate rewards to guide the generation process, making their performance sensitive to the quality of these estimates. Fig. 2 (left) reports the Spearman correlation between the intermediate (one-step prediction) and clean rewards (full

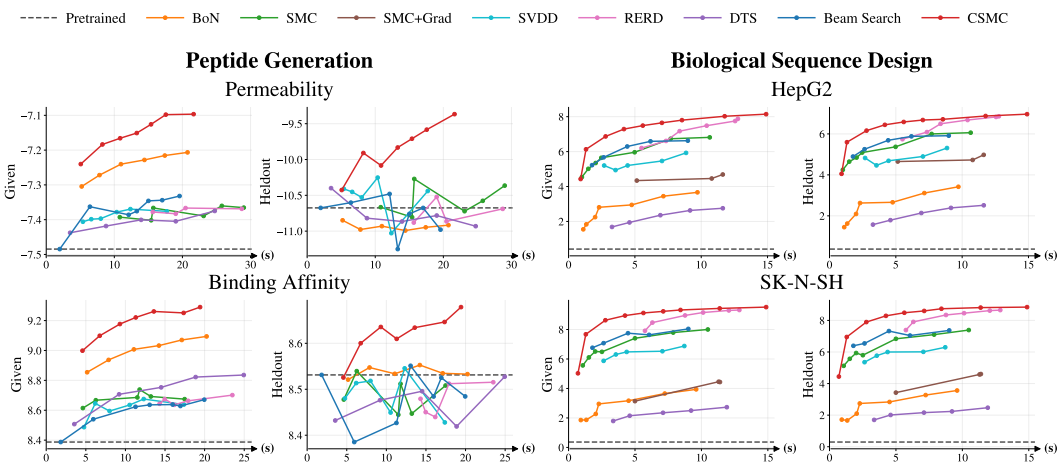

Figure 3: **Scaling results.** We show the scaling results of CSMC against baselines as we vary the wall-clock time budget for each method. For both given and helout rewards, CSMC achieves the Pareto front across peptide binding affinity and permeability, and biological sequence HepG2 and SK-N-SH activity.

rollout prediction) at each intermediate time. The Spearman correlation is a statistical measure of monotonicity, with values near one indicating strong monotonic correlation. The low correlations during early stages of diffusion, particularly for peptide generation, show that intermediate rewards provide weak, uninformative signal for guidance. In the case of peptide generation, this allows BoN, which uses only clean rewards, to outperform the other methods which rely on intermediate rewards. DTS also uses clean rewards and outperforms methods based on intermediate rewards in PepTune binding affinity, but suffers from computational inefficiency due to constructing a large search tree.

However, BoN provides no guidance during generation, so its performance depends on whether the pretrained model can generate high-reward samples on its own. For biological sequence design, the reward distribution in Fig. 2 (right) shows that high-reward samples lie in the tails of the pretrained model distribution, making BoN less effective. In these cases, it is useful to provide guidance during generation even when the intermediate rewards are imperfect. Indeed, for biological sequence design, SMC, SVDD, RERD, and beam search outperform BoN due to using guidance during generation.

CSMC combines the advantages of both methods by using clean rewards to guide the samples during generation. On therapeutic peptide generation, CSMC bypasses the inaccurate intermediate rewards, and on biological sequence design, CSMC leverages guidance through the MH transitions to find high-reward samples residing in low-density regions of the pretrained distribution.

4.3 Hyperparameter and Ablation Study

In this section, we provide hyperparameter and ablation studies on each component of CSMC. The hyperparameter and ablation studies for PepTune permeability are shown in Fig. 4

Forward Noising Time Window (Fig. 4a-b). We observe that using a time-window $[t_{\text{lo}}, t_{\text{hi}}]$ that is neither too narrow nor too wide yields the best results, giving CSMC good exploration-exploitation tradeoff. However, CSMC is not very sensitive to the time-window in general.

Number of Reverse Diffusion Steps (Fig. 4c). Using too few reverse steps (two steps) for forward-reverse diffusion results in degraded performance due to inaccurate sampling, especially as we scale compute. However, using too many reverse steps also unnecessarily increases computational cost. Using a moderate number of reverse diffusion steps (four or eight steps) achieves the best results by balancing computational cost and sampling quality.

Parallel tempering (Fig. 4 (d)). Using more parallel tempering chains improves performance due to increased exploration and parallelizability.

Temperature β (Fig. 4e). The temperature β directly impacts the target distribution p_β. Lower temperatures result in better performance due to sharpening of the reward-weighted distribution around high-reward modes. While this increases the risk of mode collapse, parallel tempering mitigates collapse through particle swaps between chains of varying temperatures.

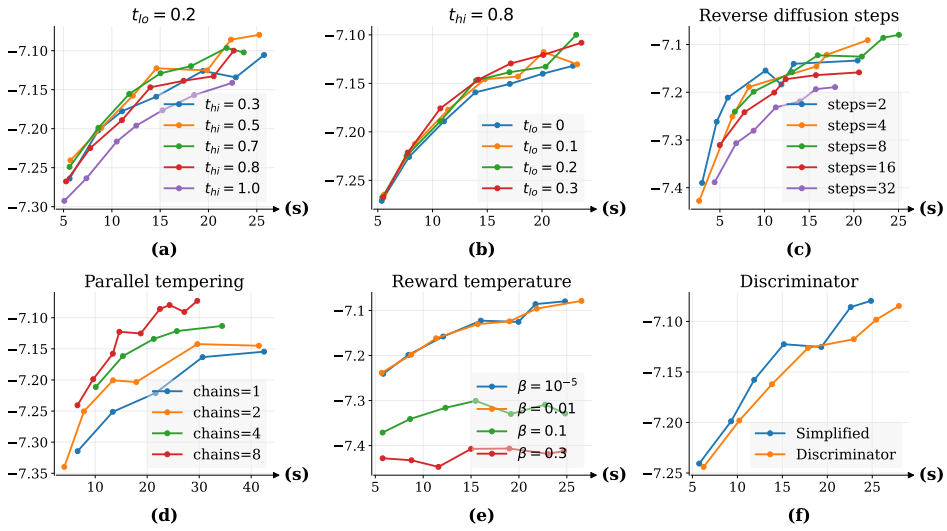

Figure 4: **Hyperparameter and ablation studies for CSMC.** We plot PepTune permeability against wall-clock time while varying **(a)** t_{hi} with $t_{\text{lo}} = 0.2$, **(b)** t_{lo} with $t_{\text{hi}} = 0.8$, **(c)** the number of reverse diffusion steps in the forward-reverse diffusion proposal, **(d)** the number of parallel-tempering chains, **(e)** the temperature β, and **(f)** the use of the simplified or discriminator acceptance probability.

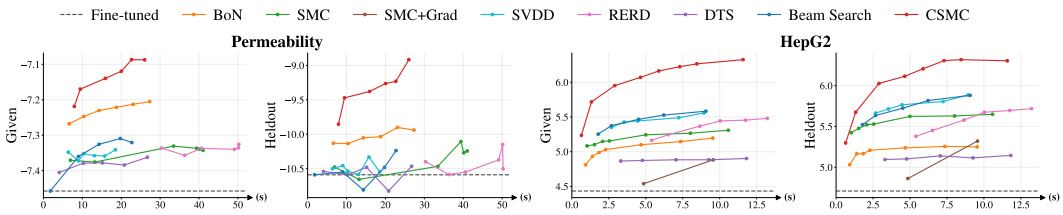

Figure 5: **Compatibility with Reward Fine-tuned Models.** We fine-tuned the pretrained diffusion model using d1 (Zhao et al., 2025), and apply each of the reward-guidance methods on top of the fine-tuned model. CSMC achieves the Pareto front among all methods.

Discriminator correction (Fig. 4f). While our main implementation uses the simplified acceptance probability in Eq. 5, it is also possible to use the discriminator to model the acceptance probability as in Eq. 2. Our simplified acceptance probability produces similar rewards while being training-free and faster since it does not require forward passes through the discriminator.

4.4 Compatibility With Reward Finetuned Models

In this section, we show the compatibility of CSMC with reward fine-tuned diffusion models. We fine-tune the diffusion models using d1 (Zhao et al., 2025), and run all methods using the fine-tuned diffusion model. As shown in Fig. 5, CSMC with the fine-tuned model shows the best scaling compared to the baselines applied on top of the fine-tuned model.

5 Conclusion

We propose **CSMC**, a novel training-free reward-guided sampler for discrete diffusion models which avoids reliance on noisy intermediate rewards by constructing a Markov Chain of clean samples via the Metropolis-Hastings algorithm. Our proposal distribution, modeled through a forward–backward combination, makes the acceptance probability tractable. Experiments on molecular and biological sequence generation with various reward functions demonstrate superior performance of our method compared to previous methods that rely on intermediate rewards.

Limitations and Future Work. While our method is effective in reward-guided sampling, mixing rate is affected by the proposal distribution. Optimizing the proposal distribution so that it is closer to the target distribution may help improve convergence and is left for future work. Lastly, designing an adaptive scheme for sampling forward noising times is a natural extension to achieve better reward-guided sampling.

AI USE STATEMENT

In this work, we used generative AI tools to aid and polish writing, aid in implementing the method, and aid in proving theorems. We have not used generative AI tools to draft entire sections, provide feedback on research, methodology or experiments, assisting with translation, interpreting results, and generating synthetic datasets, cleaning and reformatting dataset are not applicable to this work. We have reviewed all AI-assisted work. The authors have manually checked all code generated by AI, as well as the writing, and mathematical statements and proofs used in this work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work focuses on reward-guided generation at inference-time for discrete diffusion models. Although generative models could potentially be misused, this work focuses on scientific applications (e.g., peptide and DNA sequence design), which warrant no additional concerns other than those involved with generative models in general.

REPRODUCIBILITY STATEMENT

We provide the information and details needed to reproduce our theoretical and empirical results in the main paper and appendix. The complete CSMC and parallel-tempering algorithms are specified in Sec. 3 and App. A. App. A and App. B state the assumptions and proofs of convergence for our proposed algorithm. Sec. 4 and App. C describe the datasets, pretrained and reward models, public model sources, evaluation protocol, computational resources, and method hyperparameters. Code will be made publicly available after acceptance.

REFERENCES

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *NeurIPS*, 2021.

Žiga Avsec, Vikram Agarwal, Daniel Visentin, Joseph R Ledsam, Agnieszka Grabska-Barwinska, Kyle R Taylor, Yannis Assael, John Jumper, Pushmeet Kohli, and David R Kelley. Effective gene expression prediction from sequence by integrating long-range interactions. *Nature methods*, 18 (10):1196–1203, 2021.

Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Roni Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Universal guidance for diffusion models. In *ICLR*, 2024. URL `https://openreview.net/forum?id=pzpWBbnwiJ`.

Andrew Campbell, Joe Benton, Valentin De Bortoli, Thomas Rainforth, George Deligiannidis, and Arnaud Doucet. A continuous time framework for discrete denoising models. *NeurIPS*, 2022.

Wenda Chu, Zihui Wu, Yifan Chen, Yang Song, and Yisong Yue. Split gibbs discrete diffusion posterior sampling. *NeurIPS*, 2025.

Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *ICLR*, 2023.

Pierre Del Moral. *Feynman-Kac formulae: genealogical and interacting particle systems with applications*. Springer, 2004.

Arnaud Doucet, Nando De Freitas, and Neil Gordon. An introduction to sequential monte carlo methods. In *Sequential Monte Carlo methods in practice*, pp. 3–14. Springer, 2001.

Sager J Gosai, Rodrigo I Castro, Natalia Fuentes, John C Butts, Kousuke Mouri, Michael Alasoadura, Susan Kales, Thanh Thanh L Nguyen, Ramil R Noche, Arya S Rao, et al. Machine-guided design of cell-type-targeting cis-regulatory elements. *Nature*, 634(8036):1211–1220, 2024.

W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. 1970.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 2020.

Vineet Jain, Kusha Sareen, Mohammad Pedramfar, and Siamak Ravanbakhsh. Diffusion tree sampling: Scalable inference-time alignment of diffusion models. *NeurIPS*, 2025.

Jaihoon Kim, Taehoon Yoon, Jisung Hwang, and Minhyuk Sung. Inference-time scaling for flow models via stochastic generation and rollover budget forcing. *NeurIPS*, 2025a.

Sunwoo Kim, Minkyu Kim, and Dongmin Park. Test-time alignment of diffusion models without reward over-optimization. In *ICLR*, 2025b.

Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In *European conference on machine learning*, pp. 282–293. Springer, 2006.

Seul Lee, Karsten Kreis, Srimukh Prasad Veccham, Meng Liu, Danny Reidenbach, Yuxing Peng, Saee Gopal Paliwal, Weili Nie, and Arash Vahdat. Genmol: A drug discovery generalist with discrete diffusion. In *ICML*, 2025.

Xiner Li, Yulai Zhao, Chenyu Wang, Gabriele Scalia, Gokcen Eraslan, Surag Nair, Tommaso Biancalani, Shuiwang Ji, Aviv Regev, Sergey Levine, et al. Derivative-free guidance in continuous and discrete diffusion models with soft value-based decoding. *NeurIPS*, 2025.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *ICML*, 2024.

Nanye Ma, Shangyuan Tong, Haolin Jia, Hexiang Hu, Yu-Chuan Su, Mingda Zhang, Xuan Yang, Yandong Li, Tommi Jaakkola, Xuhui Jia, et al. Scaling inference time compute for diffusion models. In *CVPR*, 2025.

Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

Kevin P Murphy. *Probabilistic machine learning: Advanced topics*. MIT press, 2023.

Christian A Naesseth, Fredrik Lindsten, Thomas B Schön, et al. Elements of sequential monte carlo. *Foundations and Trends in Machine Learning*, 12(3):307–392, 2019.

Tsuneyasu Okabe, Masaaki Kawata, Yuko Okamoto, and Masuhiro Mikami. Replica-exchange monte carlo method for the isobaric–isothermal ensemble. *Chemical physics letters*, 335(5-6):435–439, 2001.

Yuta Oshima, Masahiro Suzuki, Yutaka Matsuo, and Hiroki Furuta. Inference-time text-to-video alignment with diffusion latent beam search. *NeurIPS*, 2025.

Zijing Ou, Chinmay Pani, and Yingzhen Li. Inference-time scaling of discrete diffusion models via importance weighting and optimal proposal design. In *ICLR*, 2026.

Christian P Robert and George Casella. Metropolis–hastings algorithms. In *Introducing Monte Carlo Methods with R*, pp. 167–197. Springer, 2009.

Subham Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *NeurIPS*, 2024.

Yair Schiff, Subham Sekhar Sahoo, Hao Phung, Guanghan Wang, Sam Boshar, Hugo Dalla-torre, Bernardo P de Almeida, Alexander Rush, Thomas Pierrot, and Volodymyr Kuleshov. Simple guidance mechanisms for discrete diffusion models. *ICLR*, 2025.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. *NeurIPS*, 2024.

Raghav Singhal, Zachary Horvitz, Ryan Teehan, Mengye Ren, Zhou Yu, Kathleen McKeown, and Rajesh Ranganath. A general framework for inference-time scaling and steering of diffusion models. In *ICML*, 2025.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *ICLR*, 2021.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. *NeurIPS*, 2020.

Robert H Swendsen and Jian-Sheng Wang. Replica monte carlo simulation of spin-glasses. *Physical review letters*, 57(21):2607, 1986.

Sophia Tang, Yinuo Zhang, and Pranam Chatterjee. PepTune: De novo generation of therapeutic peptides with multi-objective-guided discrete diffusion. In *ICML*, 2025.

Ibrahim I Taskiran, Katina I Spanier, Hannah Dickmänken, Niklas Kempynck, Alexandra Pančíková, Eren Can Ekşi, Gert Hulselmans, Joy N Ismail, Koen Theunis, Roel Vandepoel, et al. Cell-type-directed design of synthetic enhancers. *Nature*, 626(7997):212–220, 2024.

Masatoshi Uehara, Xingyu Su, Yulai Zhao, Xiner Li, Aviv Regev, Shuiwang Ji, Sergey Levine, and Tommaso Biancalani. Reward-guided iterative refinement in diffusion models at test-time with applications to protein and DNA design. In *ICML*, 2025.

Maxime Vono, Nicolas Dobigeon, and Pierre Chainais. Split-and-augmented gibbs sampler—application to large-scale inference problems. *IEEE Transactions on Signal Processing*, 67 (6):1648–1661, 2019.

Chenyu Wang, Masatoshi Uehara, Yichun He, Amy Wang, Avantika Lal, Tommi Jaakkola, Sergey Levine, Aviv Regev, Tommaso Biancalani, et al. Fine-tuning discrete diffusion models via reward optimization with applications to dna and protein design. In *ICLR*, 2025.

Shih-Hsin Wang, Joel Keller, Taos Transue, Drake Benjamin Brown, Thomas Strohmer, and Bao Wang. Test-time guidance for flow-based generative models via parallel tempering on source distributions. In *ICML*, 2026.

David Weininger. Smiles, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of chemical information and computer sciences*, 28(1):31–36, 1988.

Luhuan Wu, Brian Trippe, Christian Naesseth, David Blei, and John P Cunningham. Practical and asymptotically exact conditional sampling in diffusion models. *NeurIPS*, 2023.

Jiwen Yu, Yinhuai Wang, Chen Zhao, Bernard Ghanem, and Jian Zhang. Freedom: Training-free energy-guided conditional diffusion model. In *ICCV*, 2023.

Yinuo Zhang, Sophia Tang, Tong Chen, Elizabeth Mahood, Sophia Vincoff, and Pranam Chatterjee. Peptiverse: A unified platform for therapeutic peptide property prediction. *Nature Communications*, 2026.

Siyan Zhao, Devaansh Gupta, Qinqing Zheng, and Aditya Grover. d1: Scaling reasoning in diffusion large language models via reinforcement learning. *NeurIPS*, 2025.

APPENDIX

A MARKOV CHAIN MONTE CARLO

In this section, we present a brief introduction to Markov Chain Monte Carlo (MCMC) algorithms. For more details, we refer the reader to chapters 2.6 and 12 of Murphy (2023).

A.1 MARKOV CHAIN BASICS

A Markov chain is a collection of time-indexed random variables $(X_t)_{t\in\mathbb{T}}$ that satisfies the Markov property:

Definition A.1 (Markov Property)**.** *A collection of random variables $(X_t)_{t\in\mathbb{T}}$ satisfies the* **Markov Property** *if*

$$p(x_{t+\tau}|x_t, x_{1:t-1}) = p(x_{t+\tau}|x_t). \tag{6}$$

The finite-state Markov chain can be described by a ***transition matrix*** *A where $A_{ij} = p(X_t = j|X_{t-1} = i)$. We define $A_{ij}(\tau) = p(X_{t+\tau} = j|X_t = i)$*

If we continue to draw consecutive samples from the Markov chain and plot the distribution of the drawn samples, we may notice that the distribution converges to a fixed **stationary distribution**, formally defined as follows:

Definition A.2 (Stationary Distribution)**.** *A probability distribution π is a* **stationary distribution** *of a Markov chain with transition matrix A if $\pi A = \pi$, that is, π is invariant to the Markov transitions.*

The stationary distribution may not be unique for each Markov chain. The uniqueness of the stationary distribution depends on two conditions: (1) irreducibility and (2) ergodicity.

A Markov chain is **irreducible** if each state is reachable from any other state (i.e., $p(X_{t+\tau} = j|X_t = i) > 0$ for all $i, j \in \mathcal{S}$ and some τ).

Ergodicity is defined by two properties: (1) aperiodicity and (2) non-null recurrence. A state is **aperiodic** if its period $d(i) \triangleq gcd(\{t : A_{ii}(t) > 0\})$ is 1. A chain is aperiodic if all of its states are aperiodic. A state is **recurrent** if that state will be revisited with probability 1. A recurrent state is **non-null** if the expected time to return to that state is finite.

With these definitions in mind, we have a sufficient condition for the existence and uniqueness of the stationary distribution of a Markov chain:

Theorem A.1. *Every irreducible, ergodic Markov chain has a unique stationary distribution π.*

While this sufficient condition is useful, it only proves existence and uniqueness of a stationary distribution, not the form of the stationary distribution itself. To characterize the stationary distribution, we use the **detailed balance conditions**.

Definition A.3 (Detailed Balance)**.** *A Markov chain A satisfies detailed balance with respect to a distribution π if $\pi_i A_{ij} = \pi_j A_{ji}$. A Markov chain satisfying detailed balance for some distribution is said to be* **time reversible**.

It can be easily shown that a Markov chain satisfying detailed balance has a stationary distribution:

Theorem A.2. *If a Markov chain with transition matrix A satisfies the detailed balance condition with respect to a distribution π, then π is a stationary distribution of the chain.*

Proof.

$$\begin{aligned}\sum_i \pi_i A_{ij} &= \sum_i \pi_j A_{ji} \quad \text{(Detailed Balance)} \\ &= \pi_j \sum_i A_{ji} \\ &= \pi_j\end{aligned}$$

Hence, $\pi = \pi A$ □

Using Theorems A.1 and A.2, we can fully characterize the unique stationary distribution of a Markov chain.

Corollary A.2.1. *If a Markov chain with transition matrix A is irreducible, ergodic, and satisfies the detailed balance condition with respect to π, then π is its unique stationary distribution*

A.2 The Metropolis-Hastings (MH) Algorithm

In this subsection, we present a brief overview of the Metropolis-Hastings (MH) algorithm. For more details, we refer the reader to chapter 12 section 2 of Murphy (2023).

The MH algorithm constructs a Markov chain which converges to a target distribution $p^\star(x)$. In order to do so, a **proposal distribution** $q(\cdot|x)$ proposes to move from the current state x to a new state $x' \sim q(\cdot|x)$. After proposing the new state x', MH decides whether to accept or reject the new state with the acceptance probability

$$A(x'|x) = \min(1, \alpha), \quad \alpha = \frac{p^\star(x')q(x|x')}{p^\star(x)q(x'|x)}.$$

Intuitively, the accept-reject step is necessary as the proposal distribution may not match the target distribution. Samples closer to the target distribution are accepted with higher probability whereas samples further from the target distribution are rejected. The MH algorithm is summarized in Alg. 2. The Markov chain constructed by the MH algorithm has the following transition matrix:

$$p(x'|x) = \begin{cases} q(x'|x)A(x'|x) & \text{if } x' \neq x \\ q(x|x) + \sum_{x' \neq x} q(x'|x)(1 - A(x'|x)) & \text{otherwise} \end{cases} \tag{7}$$

Algorithm 2 Metropolis-Hastings Algorithm (MH)

1: Sample initial state $x^{(1)}$
2: **for** $k = 1, \ldots, K-1$ **do**
3: $\quad \tilde{x}^{(k+1)} \sim q(\cdot|x^{(k)})$
4: $\quad \alpha \leftarrow \frac{p^\star(\tilde{x}^{(k+1)})q(x^{(k)}|\tilde{x}^{(k+1)})}{p^\star(x^{(k)})q(\tilde{x}^{(k+1)}|x^{(k)})}$
5: $\quad A \leftarrow \min(1, \alpha)$
6: $\quad \rho \sim \mathcal{U}(0, 1)$
7: $\quad x^{(k+1)} \leftarrow \tilde{x}^{(k+1)}$ if $\rho < A$ else $x^{(k)}$
8: **end for**
9: $\{x^{(1)}, \ldots, x^{(K)}\}$

Theorem A.3 (Theorem 12.2.1 from Murphy (2023))**.** *If the transition matrix defined by Eq. 7 defined by the MH algorithm is ergodic and irreducible, $p^\star$ is its unique limiting distribution.*

Proof. We show that $p^\star$ satisfies the detailed balance condition. Convergence to the target stationary distribution follows by Theorem A.2.

Without loss of generality, assume $p^\star(x)q(x'|x) \geq p^\star(x')q(x|x')$. Then, $\alpha = \frac{p^\star(x')q(x|x')}{p^\star(x)q(x'|x)} \leq 1$ and thus $A(x'|x) = \alpha$. Similarly, by switching the arguments, $A(x|x') = 1$.

To move from x to x', x' must be proposed and accepted. Hence,

$$\begin{aligned} p(x'|x) &= q(x'|x)A(x'|x) && \text{from Eq. 7} \\ &= \frac{p^\star(x')q(x|x')}{p^\star(x)} \end{aligned}$$

Rearranging, $p^\star(x)p(x'|x) = p^\star(x')q(x|x')$. To show detailed balance, it suffices to prove $q(x|x') = p(x|x')$:

$$\begin{aligned} q(x|x') &= q(x|x')A(x|x') && \because A(x|x') = 1 \\ &= p(x|x') && \text{from Eq. 7} \end{aligned}$$

Since the MH Markov chain satisfies the detailed balance equation, $p^\star$ is its stationary distribution. If the chain that is ergodic and irreducible, Theorem A.1 also ensures uniqueness. □

A.3 Parallel Tempering

Running the MH algorithm at a single low temperature improves the performance but runs the risk of mode collapse. To mitigate this, parallel tempering (Swendsen & Wang, 1986) runs multiple MH chains at different temperatures in parallel. At each step, all chains are simultaneously updated using the MH update (line 7 of Alg. 2). After each update, samples are swapped across different chains using another Metropolis-Hastings step. For this swapping step, the proposal is the deterministic swap, hence the MH acceptance probability is the probability ratio of the swapped chain and the non-swapped chain:

$$\begin{aligned} A_{\text{swap}}(x_{0,\ell}, x_{0,\ell'}) &= \min\left(1, \frac{p_{\beta_\ell}(x_{0,\ell'})p_{\beta_{\ell'}}(x_{0,\ell})}{p_{\beta_{\ell'}}(x_{0,\ell'})p_{\beta_\ell}(x_{0,\ell})}\right) \\ &= \min\left(1, \frac{\exp(r(x_{0,\ell'})/\beta_\ell))p(x_{0,\ell'}) \cdot \exp(r(x_{0,\ell})/\beta_{\ell'})p(x_{0,\ell})}{\exp(r(x_{0,\ell'})/\beta_{\ell'})p(x_{0,\ell'}) \cdot \exp(r(x_{0,\ell})/\beta_\ell)p(x_{0,\ell})}\right) \\ &= \min\left(1, \exp\left((r(x_{0,\ell'}) - r(x_{0,\ell}))\left(\frac{1}{\beta_\ell} - \frac{1}{\beta_{\ell'}}\right)\right)\right). \end{aligned}$$

To parallelize the algorithm, the deterministic even-odd (DEO) swap (Okabe et al., 2001) was proposed. At every other iteration, the even states are considered to be swapped with its adjacent odd states, and vice versa. The algorithm for CSMC with parallel tempering is shown in Alg. 3.

Algorithm 3 CSMC with Parallel Tempering

Require: Temperatures $0 < \beta_1 < \cdots < \beta_L$, iterations K, reverse steps M
1: $x_{0,\ell} \sim p^{\text{pre}}(\cdot)$ independently for $\ell = 1, \ldots, L$
2: **for** $k = 1, \ldots, K$ **do**
3: **for** $\ell = 1, \ldots, L$ **do**
4: $t_{0,\ell} \sim \mathcal{U}(t_{\text{lo}}, t_{\text{hi}})$
5: $z_\ell \sim \overrightarrow{p}_{t_{0,\ell}}(\cdot \mid x_{0,\ell})$
6: $\tilde{x}_{0,\ell} \sim \overleftarrow{p}(\cdot \mid z_\ell)$ ▷ M-step reverse diffusion
7: $\alpha \leftarrow \exp\big((r(\tilde{x}_{0,\ell}) - r(x_{0,\ell}))/\beta_\ell\big)$
8: $A \leftarrow \min(1, \alpha)$
9: $\rho \sim \mathcal{U}(0, 1)$
10: $x_{0,\ell} \leftarrow \tilde{x}_{0,\ell}$ if $\rho < A$ else $x_{0,\ell}$
11: **end for**
12: $b \leftarrow 1 + ((k-1) \bmod 2)$ ▷ Alternate odd and even pairs
13: **for** $\ell = b, b+2, \ldots, L-1$ **do**
14: $\alpha_{\text{swap}} \leftarrow \exp\big((r(x_{0,\ell+1}) - r(x_{0,\ell}))(1/\beta_\ell - 1/\beta_{\ell+1})\big)$
15: $A_{\text{swap}} \leftarrow \min(1, \alpha_{\text{swap}})$
16: $\rho_{\text{swap}} \sim \mathcal{U}(0, 1)$
17: **if** $\rho_{\text{swap}} < A_{\text{swap}}$ **then**
18: $(x_{0,\ell}, x_{0,\ell+1}) \leftarrow (x_{0,\ell+1}, x_{0,\ell})$ ▷ DEO swap
19: **end if**
20: **end for**
21: **end for**
22: **return** $x_{0,1}^{(K)}$ ▷ Return sample from lowest-temperature chain

A.4 Irreducibility and Ergodicity of CSMC

In the main text, we have explained how CSMC is derived from the MH algorithm. Thus, the detailed balance conditions is satisfied with respect to p_β, and p_β is a stationary distribution for the Markov chain constructed by CSMC. We now verify that the Markov chain constructed by CSMC is irreducbile and ergodic, which shows that p_β is the unique stationary distribution of CSMC by Theorem A.1. First, we restrict the domain of the target distribution:

$$\mathcal{S} := \text{supp}(p^{\text{pre}}) \quad \text{and} \quad p_\beta(x) := \frac{1}{Z_\beta}\exp(r(x)/\beta)p^{\text{pre}}(x), \qquad x \in \mathcal{S},$$

where $\beta > 0$, $r(x)$ is finite on $\mathcal{S}$, and Z_β is the normalizing constant. We restrict the chain to $\mathcal{S}$, since states outside the support of the pretrained model also have zero probability under p_β. Under these conditions, we can prove irreducibility and ergodicity:

Theorem A.4 (Irreducibility and ergodicity of CSMC)**.** *Suppose that $\mathcal{X}$ is finite. Let ν denote the distribution used to sample the noising time t. Assume that there exist a set of times $\mathcal{T}_\star$ with $\nu(\mathcal{T}_\star) > 0$ and a noisy state $z_\star$ such that*

$$\overrightarrow{p}_t(z_\star|x) > 0 \qquad \textit{for every } x \in \mathcal{S} \textit{ and } t \in \mathcal{T}_\star. \tag{8}$$

Then the CSMC *chain in Alg. 1 is irreducible and ergodic (aperiodic and positive recurrent) on $\mathcal{S}$.*

Proof. The forward–reverse proposal kernel can be written as

$$q(y|x) = \int \nu(\mathrm{d}t) \sum_z \overrightarrow{p}_t(z|x) \overleftarrow{p}_t(y|z). \tag{9}$$

The matching marginals condition and Bayes' rule imply (refer to Sec. B.1 for more details)

$$p^{\text{pre}}(x) \overrightarrow{p}_t(z|x) = p^{\text{pre}}_t(z) \overleftarrow{p}_t(x|z). \tag{10}$$

Therefore the proposal is reversible with respect to the pretrained distribution:

$$\begin{aligned} p^{\text{pre}}(x) q(y|x) &= \int \nu(\mathrm{d}t) \sum_z p^{\text{pre}}_t(z) \overleftarrow{p}_t(x|z) \overleftarrow{p}_t(y|z) \\ &= \int \nu(\mathrm{d}t) \sum_z p^{\text{pre}}(y) \overrightarrow{p}(z|y) \overleftarrow{p}_t(x|z) \\ &= p^{\text{pre}}(y) q(x|y). \end{aligned}$$

Fix any $x, y \in \mathcal{S}$. For $t \in \mathcal{T}_\star$, rearranging Eq. 10 gives

$$\overleftarrow{p}_t(y|z_\star) = \frac{p^{\text{pre}}(y) \overrightarrow{p}_t(z_\star|y)}{p^{\text{pre}}_t(z_\star)} > 0.$$

The proposal probability $\overrightarrow{p}_t(z_\star|x) \overleftarrow{p}_t(y|z_\star)$ is strictly positive, and Eq. 9 yields

$$q(y|x) \geq \int_{\mathcal{T}_\star} \nu(\mathrm{d}t)\, \overrightarrow{p}_t(z_\star|x) \overleftarrow{p}_t(y|z_\star) > 0.$$

Moreover, p_β is strictly positive on $\mathcal{S}$, so the MH acceptance probability from x to y is also strictly positive. Hence the probability of transitioning from x to y is strictly positive for every $x, y \in \mathcal{S}$, proving irreducibility in one step.

Taking $y = x$ in the same argument gives $q(x|x) > 0$. A proposal equal to the current state is accepted with probability one, so the probability of remaining at the same state is strictly positive for every $x \in \mathcal{S}$. The chain is therefore aperiodic. Finally, every irreducible Markov chain on a finite state space is positive recurrent. Hence, the chain is ergodic. □

Discrete diffusion models have finite vocabulary and sequence length, hence $\mathcal{X}$ is finite. Also note that condition 8 is satisfied by MDMs: take $z_\star$ to be the fully masked sequence and any nonzero noise level at which every token has positive probability of being masked. Thus, it suffices to choose $t_{\text{lo}} > 0$.

Remark 2. While Theorem A.4 shows uniqueness of the stationary distribution of CSMC, the proof requires finiteness of $\mathcal{X}$ (otherwise positive recurrence does not immediately follow from irreducibility without further assumptions). Finiteness of $\mathcal{X}$ always holds for discrete diffusion models but may not hold for continuous diffusion models.

B ACCEPTANCE PROBABILITY

The cancellation of the acceptance probability relies on the assumption that

$$p^{\text{pre}}(x_0)\overrightarrow{p}(x_t|x_0) = p_t^{\text{pre}}(x_t)\overleftarrow{p}(x_0|x_t). \tag{11}$$

In Sec. B.1 below, we show that Eq. 11 is enforced by the standard diffusion objectives.

In Sec. B.2, we also show that $\chi_{x_t}(x_0'|x_0)$ can be exactly obtained via a trained discriminator. Finally, we show that using the exact discriminator-corrected acceptance probability yields almost the same performance as our simplified acceptance probability. Thus in practice, it is sufficient to use our simplified acceptance probability in Eq. 5.

B.1 CANCELLATION ASSUMPTION

Bayes' theorem gives

$$\begin{aligned}\overrightarrow{p}(x_0|x_t) &= \frac{p^{\text{pre}}(x_0)\overrightarrow{p}(x_t|x_0)}{\sum_{\tilde{x}_0} p^{\text{pre}}(\tilde{x}_0)\overrightarrow{p}(x_t|\tilde{x}_0)} \\ &= \frac{p^{\text{pre}}(x_0)\overrightarrow{p}(x_t|x_0)}{p_t^{\text{pre}}(x_t)},\end{aligned}$$

where $p_t^{\text{pre}}(x_t) := \sum_{\tilde{x}_0} p^{\text{pre}}(\tilde{x}_0)\overrightarrow{p}(x_t|\tilde{x}_0)$.

Rearranging, we have

$$p^{\text{pre}}(x_0)\overrightarrow{p}(x_t|x_0) = p_t^{\text{pre}}(x_t)\overrightarrow{p}(x_0|x_t). \tag{12}$$

Comparing this with Eq. 11, we require the following matching marginals assumption for cancellation to exactly hold:

$$\overleftarrow{p}(x_0|x_t) = \overrightarrow{p}(x_0|x_t). \tag{13}$$

We now show that this equality is enforced by standard diffusion training objectives.

Lemma B.1. *Let $I(X;Y|Z)$ denote the mutual information of X and Y conditioned on Z. Then,*

$$\mathbb{E}_{p(x_0,x_t)} D_{\text{KL}}(\overrightarrow{p}(x_s|x_0,x_t)\|\overleftarrow{p}(x_s|x_t)) = I(x_s;x_0|x_t) + \mathbb{E}_{p_t(x_t)} D_{\text{KL}}(\overrightarrow{p}(x_s|x_t)\|\overleftarrow{p}(x_s|x_t).$$

Proof.

$$\begin{aligned}\mathbb{E}_{p(x_0,x_t)} D_{\text{KL}}(\overrightarrow{p}(x_s|x_0,x_t)\|\overleftarrow{p}(x_s|x_t)) &= \int_{\mathcal{X}^3} p(x_0,x_t)\overrightarrow{p}(x_s|x_0,x_t)\log\left(\frac{\overrightarrow{p}(x_s|x_0,x_t)}{\overleftarrow{p}(x_s|x_t)}\right)\mathrm{d}x_s\,\mathrm{d}x_t\,\mathrm{d}x_0 \\ &= \mathbb{E}_{p(x_0,x_s,x_t)}\log\frac{\overrightarrow{p}(x_s|x_0,x_t)}{\overrightarrow{p}(x_s|x_t)} + \mathbb{E}_{p(x_0,x_s,x_t)}\log\frac{\overrightarrow{p}(x_s|x_t)}{\overleftarrow{p}(x_s|x_t)}.\end{aligned}$$

The first term can be rewritten as

$$\begin{aligned}\mathbb{E}_{p(x_0,x_s,x_t)}\log\frac{\overrightarrow{p}(x_s|x_0,x_t)}{\overrightarrow{p}(x_s|x_t)} &= \mathbb{E}_{p(x_0,x_s,x_t)}\log\frac{\overrightarrow{p}(x_s,x_0|x_t)}{\overrightarrow{p}(x_0|x_t)\overrightarrow{p}(x_s|x_t)} \\ &= I(x_s;x_0|x_t).\end{aligned}$$

Hence,

$$\mathbb{E}_{p(x_0,x_t)} D_{\text{KL}}(\overrightarrow{p}(x_s|x_0,x_t)\|\overleftarrow{p}(x_s|x_t)) = I(x_s;x_0|x_t) + \mathbb{E}_{p_t(x_t)} D_{\text{KL}}(\overrightarrow{p}(x_s|x_t)\|\overleftarrow{p}(x_s|x_t)).$$

□

Lemma B.1 shows that minimizing the diffusion objective implicitly matches the forward and reverse processes of adjacent steps. Since many diffusion models (MDM (Sahoo et al., 2024), USM (Schiff et al., 2025), Gaussian diffusion models (Ho et al., 2020)) are trained using this objective, a perfectly

trained diffusion model exactly matches the transition probabilities for each consecutive step, which implies

$$\begin{aligned}\overleftarrow{p}(x_0|x_t) &= \int_{X_{1:t-1}} \prod_{s=1}^{t} \overleftarrow{p}(x_{s-1}|x_s)\,\mathrm{d}x \\ &= \int_{X_{1:t-1}} \prod_{s=1}^{t} \overrightarrow{p}(x_{s-1}|x_s)\,\mathrm{d}x \\ &= \overrightarrow{p}(x_0|x_t)\end{aligned}$$

Using a perfectly trained denoiser and matching samplers guarantees Eq. 11, while using an imperfect denoiser or different sampler may break this guarantee. However, we show in Sec. B.2 that using an imperfect denoiser and few-step sampling does not yield significantly different performance.

B.2 Discriminator Training

In this section, we provide more details on the discriminator.

First, we prove Theorem 3.1 from Sec. 3.3 to show how we can obtain $\chi_{x_t}(x_0'|x_0)$ from the discriminator.

Theorem B.2 (Sec. 3.3, Theorem 3.1). *Let $D^\star : \mathcal{X}^3 \times \mathbb{R} \to [0,1]$ be the optimal discriminator trained on the positive and negative quadruplets. Then,*

$$\chi_{x_t}(x_0'|x_0) = \log\left(\frac{D^\star(x_0, x_0', x_t, t)}{1 - D^\star(x_0, x_0', x_t, t)}\right) \tag{14}$$

Proof. The discriminator is trained by maximizing the following objective:

$$\begin{aligned}V(D) &= \mathbb{E}_{(x_0,x_0',x_t,t)\sim p^+}\Big[\log D(x_0, x_0', x_t, t) + \log(1 - D(x_0', x_0, x_t, t))\Big] \\ &= \int_{\mathbb{X}} p^+(\mathrm{d}x_0, \mathrm{d}x_0', \mathrm{d}x_t, \mathrm{d}t)\Big(\log D(x_0, x_0', x_t, t) + \log(1 - D(x_0', x_0, x_t, t))\Big)\,\mathrm{d}\mathbb{X} \\ &= \int_{\mathbb{X}} p^+(\mathrm{d}x_0, \mathrm{d}x_0', \mathrm{d}x_t, \mathrm{d}t)\log D(x_0, x_0', x_t, t) \\ &\quad + p^-(\mathrm{d}x_0, \mathrm{d}x_0', \mathrm{d}x_t, \mathrm{d}t)\log(1 - D(x_0, x_0', x_t, t)),\end{aligned}$$

where $\mathbb{X} = \mathcal{X}^3 \times \mathbb{R}$. The third line follows by making the substitution $(x_0, x_0', x_t, t) \mapsto (x_0', x_0, x_t, t)$ for the second term. For any $(a, b) \in \mathbb{R}^2 \setminus \{0, 0\}$, the function $f(y) : [0,1] \to \mathbb{R}$ defined by the mapping $y \mapsto a\log(y) + b\log(1-y)$ is maximized at $\frac{a}{a+b}$. Setting $a = p^+(x_0, x_0', x_t, t)$ and $b = p^-(x_0, x_0', x_t, t)$, the maximizer of $V(D)$ is

$$D^\star(x_0, x_0', x_t, t) = \frac{p^+(x_0, x_0', x_t, t)}{p^+(x_0, x_0', x_t, t) + p^-(x_0, x_0', x_t, t)},$$

Then,

$$\begin{aligned}\chi_{x_t}(x_0'|x_0) &= \log\left(\frac{p^{\text{pre}}(x_0')\overrightarrow{p}(x_t|x_0')\overleftarrow{p}(x_0|x_t)}{p^{\text{pre}}(x_0)\overrightarrow{p}(x_t|x_0)\overleftarrow{p}(x_0'|x_t)}\right) \\ &= \log\left(\frac{p^+(x_0, x_0', x_t, t)}{p^-(x_0, x_0', x_t, t)}\right) \\ &= \log\left(\frac{D^\star(x_0, x_0', x_t, t)}{1 - D^\star(x_0, x_0', x_t, t)}\right)\end{aligned}$$

□

Using the optimal discriminator, we can sample from CSMC with the exact acceptance probability. The discriminator is implemented as a lightweight neural network head that outputs the classification

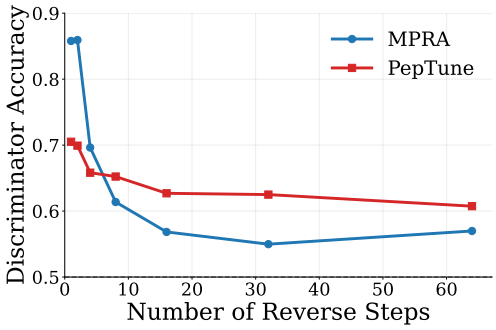

Figure B.1: **Discriminator accuracy for different numbers of reverse steps.** We report the classification accuracy as the number of reverse-diffusion steps M varies. Accuracy decreases as M increases, indicating that the original and proposed clean samples become harder to distinguish with more reverse steps.

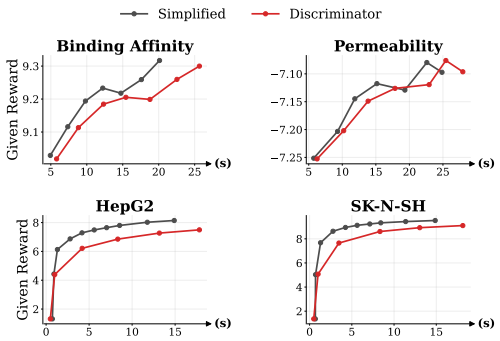

Figure B.2: **Comparison of simplified and discriminator-corrected acceptance probability.** Using the discriminator-corrected acceptance probability does not significantly affect the reward achieved by CSMC using our simplified acceptance probability.

logit. The inputs of the discriminator are the two clean samples, one noisy sample, and the timestep t. Instead of feeding the one-hot encoding of the clean and noisy samples, we feed the last hidden state of the denoising network corresponding to the samples instead, which we found significantly improved performance. To make sure the discriminator can handle varying number of reverse diffusion steps, we also input the number of reverse diffusion steps M used for the forward-reverse diffusion process.

In Fig. B.1, we plot the discriminator accuracy when using different number of reverse steps for the forward-reverse diffusion process. As shown, the discriminator has higher accuracy for fewer reverse steps, indicating that when using fewer reverse steps, the forward-reverse diffusion process changes the resulting sample in a meaningful way. However, as the number of reverse steps increases, the discriminator accuracy drops, indicating similar samples before and after applying forward-reverse diffusion.

We compare CSMC using the simplified acceptance probability with CSMC using the discriminator in Fig. B.2. As shown, using the discriminator does not significantly increase the reward while being slower due to additional discriminator forward passes.

C Experiment Details

In this section, we provide details on our experiments, including the pretrained models used and the hyperparameters of each method. All experiments were run using NVIDIA RTX3090 GPUs.

C.1 Pretrained Models

Therapeutic Peptide Generation. We use the pretrained PepTune model (Tang et al., 2025), a masked discrete diffusion model with a RoFormer denoising backbone trained on approximately 11 million peptide SMILES. The model and tokenizer implementation, model card, and instructions for obtaining the pretrained checkpoint are publicly available in the ChatterjeeLab/PepTune Hugging Face repository.

Biological Sequence Design. We train a masked discrete diffusion model on the approximately 700,000 DNA sequences from the MPRA dataset of Gosai et al. (2024). The denoiser is a four-block convolutional network with 128 hidden channels. The vocabulary used is $\{\texttt{A}, \texttt{C}, \texttt{G}, \texttt{T}, \texttt{[MASK]}\}$. The generative checkpoint used in our experiments does not currently have a public Hugging Face repository. The activity predictors are trained on disjoint data splits for given and held-out reward computation.

C.2 Reward-Guided Sampling

We use the same settings for all rewards within each domain. The main scaling parameter of each method (e.g., the number of BoN samples, SMC particles, or CSMC iterations) is varied to obtain each scaling plot.

For all experiments, we generate 128 samples and report the mean reward. The wall-clock time is obtained by running each method to generate one sample. Three timing runs are executed and the minimum run-time for each method is reported.

Best-of-N (BoN). For both tasks, we scale the number of independently generated candidates N to obtain the scaling plot.

Sequential Monte Carlo (SMC). For both tasks, the particle count N is scaled, and the reward temperature is set to $\beta = 0.1$. While using lower temperatures is possible, we find that setting the temperature too low results in degraded performance due to mode collapse. Following Kim et al. (2025b), we use an initial temperature of $\beta = 1.0$ and apply tempering to decrease the temperature to the target temperature $\beta = 0.1$ over time. We use ESS resampling with a normalized ESS threshold of 0.5. In both tasks, we return the highest-reward final particle.

Gradient-Guided Sequential Monte Carlo (SMC+Grad). We only evaluate SMC+Grad on biological sequence design because the MPRA reward model is differentiable, whereas the peptide rewards involve non-differentiable tokenization and detokenization operations. We use a temperature of $\beta = 0.1$, while the Gumbel-Softmax gradient estimator uses temperature 1.0 and one gradient estimate per particle. We return the highest-reward final particle as done for SMC.

Soft Value-Based Decoding (SVDD). We set the temperature to $\beta = 10^{-5}$, following the recommended setting from Li et al. (2025). The number of particles proposed at each step is varied to obtain the scaling plot.

Reward-guided Evolutionary Refinement in Diffusion models (RERD). For both peptide generation and biological sequence design, we maintain a global population of five samples and run generation with four SVDD proposals per particle. We set the temperature to $\beta = 0.2$ and noise each resampled clean sample back to normalized time $t = 0.1$. We vary the number of refinement iterations to obtain the scaling plot.

Diffusion Tree Sampling (DTS). The number of search iterations is scaled to obtain the scaling plots. We set the temperature to $\beta = 0.1$. We return one sample per tree.

Beam Search. We set a beam width of 4 and vary the number of proposals per sample to obtain the scaling plot.

CSMC. For peptide generation, we use 8 parallel-tempering chains with temperatures linearly spaced from 1 to 10^{-5}, and set the forward noising time window to $[0.2, 0.5]$ and number of reverse steps to eight. For biological sequence design, we use 4 parallel-tempering chains with temperatures logarithmically spaced from 1 to 10^{-5}, and set the noising time window to $[0.2, 0.6]$ with one reverse diffusion step. In both tasks, the number of iterations is varied to obtain the scaling plot.

D Additional Metrics

Since reward-guided samplers may be prone to overoptimize the reward at the expense of sample quality or naturalness, we report additional sample diversity and quality metrics in Sec. D.1, showing that CSMC does not degrade the diversity and quality of the generated samples.

In Sec. D.2, we report the target chain acceptance rate and inter-chain swap acceptance rate for CSMC, showing that acceptance rate remains sufficiently high and CSMC does not collapse to a single sample.

Method	NFE	Validity ↑	Uniqueness ↑	Diversity ↑	Reward ↑
Pretrained	128	0.305	1.000	0.641	8.387 ± 0.735
BoN	10240	1.000	1.000	0.629	9.094 ± 0.271
SMC	5120	0.977	1.000	0.648	8.674 ± 0.657
SVDD	3328	0.492	1.000	0.671	8.638 ± 0.727
RERD	4800	0.992	1.000	0.646	8.685 ± 0.610
DTS	3456	1.000	1.000	0.646	8.836 ± 0.458
Beam Search	4608	0.555	1.000	0.666	8.671 ± 0.940
CSMC	7552	1.000	1.000	0.642	9.290 ± 0.248

Table D.1: **PepTune Binding Affinity quality metrics at each method's final NFE.** We report the validity, uniqueness, diversity, and 95% confidence interval of the reward. Validity is the fraction of generated samples that decode to valid peptide SMILES, uniqueness is the fraction of valid samples that are distinct, and diversity is their mean pairwise Tanimoto distance based on the Morgan2 fingerprints. Higher is better for all metrics. CSMC achieves perfect validity and uniqueness while achieving the high diversity and the highest reward.

Method	NFE	Validity ↑	Uniqueness ↑	Diversity ↑	Reward ↑
Pretrained	128	0.305	1.000	0.641	-7.485 ± 0.264
BoN	10240	1.000	1.000	0.664	-7.207 ± 0.155
SMC	5120	0.953	1.000	0.655	-7.360 ± 0.289
SVDD	3328	0.531	1.000	0.669	-7.376 ± 0.240
RERD	4800	0.992	1.000	0.652	-7.340 ± 0.271
DTS	3456	1.000	1.000	0.652	-7.375 ± 0.288
Beam Search	4608	0.547	1.000	0.670	-7.332 ± 0.269
CSMC	7552	1.000	1.000	0.684	-7.097 ± 0.202

Table D.2: **PepTune Permeability quality metrics at each method's final NFE.** We report the validity, uniqueness, diversity, and 95% confidence interval of the reward. Validity is the fraction of generated samples that decode to valid peptide SMILES, uniqueness is the fraction of valid samples that are distinct, and diversity is their mean pairwise Tanimoto distance based on the Morgan2 fingerprints. Higher is better for all metrics. CSMC achieves perfect validity and uniqueness while achieving the highest diversity and reward.

D.1 Diversity and Quality

Therapeutic peptide generation. We report validity, uniqueness, and diversity, and the 95% confidence interval of the reward of each method's largest NFE in Tab. D.1 and Tab. D.2. Results on the model fine-tuned on permeability are included in Tab. D.3. Validity measures the fraction of generated samples that decode to valid peptides, uniqueness measures the fraction of non-duplicate valid samples, and diversity is the average pairwise distance between valid samples. The distance is defined as Tanimoto similarity of the Morgan2 fingerprints, subtracted from one.

As shown in Tab. D.1 and Tab. D.2, the pretrained model mostly generates invalid SMILES strings. Applying reward-guidance improves the validity but still fails in cases such as SVDD and beam search. Notably, BoN, DTS, and CSMC achieve perfect validity. All methods achieve perfect uniqueness. CSMC additionally achieves the highest diversity for permeability and, for binding affinity, it achieves similar diversity to the pretrained model and more diversity than BoN. As shown in Tab. D.3, CSMC can also be applied on top of the fine-tuned model, achieving the highest diversity and reward.

Biological sequence design. We report validity, uniqueness, and diversity, the 95% confidence interval of the reward, and other quality metrics for each method's largest NFE in Tab. D.4 and Tab. D.5. Results on the HepG2 fine-tuned model are shown in Tab. D.6. Diversity is calculated as the pairwise hamming distance of the sequences. ATAC accuracy is another measure for cell line activity. JASPAR correlation motifs and 3-mer frequencies measure how closely the sequences

Method	NFE	Validity	Uniqueness	Diversity	Reward
Fine-tuned	128	0.422	1.000	0.646	-7.457 $\pm$ 0.236
BoN	10240	1.000	1.000	0.651	-7.205 $\pm$ 0.133
SMC	5120	0.977	1.000	0.634	-7.331 $\pm$ 0.202
SVDD	3328	0.719	1.000	0.648	-7.341 $\pm$ 0.225
RERD	4800	1.000	1.000	0.638	-7.340 $\pm$ 0.203
DTS	3456	1.000	1.000	0.640	-7.362 $\pm$ 0.222
Beam Search	4608	0.641	1.000	0.657	-7.317 $\pm$ 0.204
CSMC	7552	1.000	1.000	0.666	-7.087 $\pm$ 0.183

Table D.3: **Fine-tuned PepTune Permeability quality metrics at each method's final NFE.** Although the fine-tuned model's mean (valid) reward only slightly higher than the pretrained model's reward, we note that the validity significantly increased. CSMC achieves perfect validity and uniqueness while achieving the highest diversity and reward.

Method	NFE	Uniqueness↑	Diversity↑	Reward↑	ATAC Acc↑	JASPAR Corr↑	3-mer Corr↑
Pretrained	100	1.000	0.748	0.065 $\pm$ 1.886	0.023	-0.093	0.012
BoN	8192	1.000	0.747	3.414 $\pm$ 1.832	0.250	0.720	0.758
SMC	8192	1.000	0.739	6.260 $\pm$ 2.374	0.469	0.879	0.787
SMC+Grad	1600	1.000	0.486	4.904 $\pm$ 1.494	0.016	0.878	0.473
SVDD	4096	1.000	0.742	5.549 $\pm$ 2.466	0.266	0.820	0.719
RERD	6144	1.000	0.735	7.876 $\pm$ 1.828	0.828	0.876	0.888
DTS	2560	1.000	0.749	2.496 $\pm$ 2.202	0.133	0.423	0.547
Beam Search	4096	1.000	0.739	6.061 $\pm$ 2.332	0.352	0.871	0.749
CSMC	10400	1.000	0.731	8.144 $\pm$ 0.966	0.781	0.904	0.904

Table D.4: **HepG2-guided MPRA sequence metrics at each method's final NFE.** We report the uniqueness, diversity, reward 95% confidence interval, and additional DNA sequence quality metrics. Uniqueness is the fraction of distinct sequences, and diversity is the mean pairwise Hamming distance. ATAC accuracy is another measure for cell line activity. JASPAR correlation motifs and 3-mer frequencies measure how closely the sequences resemble natural enhancers. Higher is better for all metrics. CSMC achieves perfect uniqueness and comparable diversity to the pretrained model, while attaining significantly higher reward and quality metrics.

resemble natural enhancer DNA sequences. Every method achieves perfect uniqueness, and CSMC retains diversity close to the pretrained model. In contrast, SMC+Grad substantially reduces diversity to 0.486 and 0.496. CSMC obtains the highest activity on both rewards. CSMC also achieves high ATAC accuracy as well as the highest JASPAR and 3-mer correlation among all methods, suggesting that we attain reward improvements without compromising sample quality and naturalness. As shown in Tab. D.6, the fine-tuned model displays collapsed diversity, which is typical for RL fine-tuned models. Despite this, CSMC can also be applied on top of the fine-tuned model to achieve the highest reward while also increasing diversity.

D.2 Acceptance Rate

We report the acceptance rate from three independent CSMC runs in Tab. D.7. The target chain accepts between 16.7% and 20.6% of the proposals, which is sufficiently high. Swap acceptance is generally high across the warmer chains. The lowest swap rate occurs between chains 1 and 2 (3.9%–7.8%), indicating infrequent exchange with the target chain. However, this swap rate is still sufficient to mitigate mode collapse.

Method	NFE	Uniqueness↑	Diversity↑	Reward↑	ATAC Acc↑	JASPAR Corr↑	3-mer Corr↑
Pretrained	100	1.000	0.748	0.042 ± 1.926	0.023	-0.151	-0.073
BoN	8192	1.000	0.746	3.458 ± 2.482	0.188	0.723	0.727
SMC	8192	1.000	0.745	7.698 ± 2.788	0.273	0.779	0.597
SMC+Grad	1600	1.000	0.496	4.373 ± 1.952	0.016	0.846	0.466
SVDD	4096	1.000	0.745	6.564 ± 3.546	0.195	0.733	0.508
RERD	6144	1.000	0.738	9.322 ± 1.089	0.820	0.795	0.864
DTS	2560	1.000	0.749	2.450 ± 2.286	0.109	0.424	0.485
Beam Search	4096	1.000	0.744	7.704 ± 2.834	0.391	0.806	0.602
CSMC	10400	1.000	0.731	9.518 ± 0.424	0.945	0.916	0.914

Table D.5: **SK-N-SH-guided MPRA sequence metrics at each method's final NFE.** We report the uniqueness, diversity, reward 95% confidence interval, and additional DNA sequence quality metrics. Uniqueness is the fraction of distinct sequences, and diversity is the mean pairwise Hamming distance. ATAC accuracy is another measure for cell line activity. JASPAR correlation motifs and 3-mer frequencies measure how closely the sequences resemble natural enhancers. Higher is better for all metrics. CSMC achieves perfect uniqueness and comparable diversity to the pretrained model, while attaining significantly higher reward and quality metrics.

Method	NFE	Uniqueness↑	Diversity↑	Reward↑	ATAC Acc↑	JASPAR Corr↑	3-mer Corr↑
Fine-tuned	100	1.000	0.350	4.433 ± 0.791	0.000	0.819	0.594
BoN	8192	1.000	0.387	5.196 ± 0.283	0.000	0.840	0.609
SMC	8192	1.000	0.373	5.309 ± 0.481	0.000	0.847	0.636
SMC+Grad	1600	1.000	0.665	5.207 ± 1.456	0.125	0.693	0.808
SVDD	4096	1.000	0.384	5.557 ± 0.559	0.008	0.868	0.644
RERD	6144	1.000	0.410	5.480 ± 0.780	0.000	0.860	0.644
DTS	2560	1.000	0.360	4.902 ± 0.411	0.000	0.830	0.603
Beam Search	4096	1.000	0.376	5.583 ± 0.542	0.000	0.863	0.635
CSMC	8192	1.000	0.467	6.325 ± 0.264	0.023	0.849	0.643

Table D.6: **Fine-tuned HepG2-guided MPRA sequence metrics at each method's final NFE.** While fine-tuning significantly improved the reward, it comes at the cost of significantly lower diversity. As a result, applying each reward-guidance method on top of the fine-tuned model does not necessarily outperform applying them on top of the pretrained models. CSMC achieves perfect uniqueness and sufficient diversity, with high reward and quality metrics.

Run	**Target chain acceptance rate**	**Swap acceptance rate**						
		$7 \leftrightarrow 8$	$6 \leftrightarrow 7$	$5 \leftrightarrow 6$	$4 \leftrightarrow 5$	$3 \leftrightarrow 4$	$2 \leftrightarrow 3$	$1 \leftrightarrow 2$
Run 1	0.186	0.353	0.765	0.922	0.922	0.412	0.843	0.078
Run 2	0.206	0.902	0.941	0.980	0.941	0.863	0.549	0.039
Run 3	0.167	0.451	0.745	0.882	0.922	0.824	0.706	0.078

Table D.7: **CSMC acceptance rate.** We report the acceptance rate of the target chain (with temperature $\beta = \beta_1$) as well as the acceptance rate for the parallel tempering swaps between adjacent chains. The target chain's acceptance rate remains sufficiently high.