Importance-Weighted Training of Diffusion Samplers

Sanghyeok Choi [1] **Sarthak Mittal** [2 3] **Víctor Elvira** [4] **Jinkyoo Park** [1] **Nikolay Malkin** [4]

Abstract

We propose an importance-weighted training framework for diffusion samplers – diffusion models trained to sample from a Boltzmann distribution – that leverages Monte Carlo methods with off-policy training to improve training efficiency and mode coverage. Building upon past attempts to use experience replay to guide the training of denoising models as policies, we derive a way to combine historical samples with adaptive importance weights so as to make the training samples better approximate the desired distribution even when the sampler is far from converged. On synthetic multi-modal targets and the Boltzmann distribution of alanine dipeptide conformations, we demonstrate improvements in distribution approximation and training stability compared to existing baselines. Our results are a step towards combining the strengths of amortized (RL- and control-based) approaches to training diffusion samplers with those of Monte Carlo methods.

1. Introduction

The problem of sampling from high-dimensional unnormalized probability distributions arises in many domains, including Bayesian inference of statistical model parameters and sampling of molecular conformations in computational chemistry (Noé et al., 2019; Holdijk et al., 2023). Solving this problem requires exploration of a complex, multimodal energy landscape defining a Boltzmann distribution that is intractable to sample from directly. Classical approaches such as Monte Carlo methods – including both fast Markov Chain Monte Carlo (MCMC) methods like Hamiltonian Monte Carlo (HMC; Duane et al., 1987; Hoffman et al., 2014) and accelerated Langevin dynamics (Leimkuhler et al., 2014) and particle-based methods like Sequential Monte Carlo (SMC; Halton, 1962; Del Moral et al., 2006) – have been used to address this challenge for decades.

Recent work has considered amortized methods, in which a parametric model, such as a neural network, is trained to directly sample from the target distribution (Noé et al., 2019). In particular, diffusion models have shown promise as the generative models used for amortized sampling: in contrast to diffusion models trained to maximize a variational bound on log-likelihood of a dataset (Ho et al., 2020; Song et al., 2021b;a), these *diffusion samplers* are trained, by one of many available objectives, assuming access to a queryable energy function (see related work in §4).

In contrast to Monte Carlo methods without learnable components, amortized samplers, including diffusion samplers, can take advantage of the generalization abilities of deep networks to exploit regularities in the target distribution. In addition, while Monte Carlo methods typically require a large number of steps or samples to produce unbiased samples, amortized samplers can generate samples in finite time once trained. On the other hand, the anytime nature of Monte Carlo methods can be seen as a strength: they are guaranteed to approach the target distribution as the number of steps or particles increases, while amortized samplers may fail to converge to the target distribution due to insufficient model capacity or optimization issues, such as mode collapse or overfitting to a subset of the target distribution.

In this work, we propose an importance-weighted training framework for diffusion samplers that combines the strengths of amortized sampling with Monte Carlo methods. We first derive a connection between diffusion samplers and importance sampling that allows us to use an imperfectly trained sampler as a proposal distribution. We then introduce a method for using importance-weighted samples obtained from past iterations of a diffusion sampler to improve the training of the current model, which we interpret as a form of prioritized experience replay. We also introduce an adaptive importance weight tempering strategy, which reduces variance in the learning signal and significantly improves training stability.

We demonstrate the effectiveness of our method on synthetic targets and on the Boltzmann distribution of alanine dipeptide conformations, finding improvements in target distribution approximation over comparable prior work.

[1]KAIST [2]Université de Montreal [3]Mila – Québec AI Institute [4]University of Edinburgh. Correspondence to: Sanghyeok Choi <sanghyeok.choi@kaist.ac.kr>, Nikolay Malkin <nmalkin@ed.ac.uk>.

Proceedings of the Workshop on Generative AI for Biology at the 42nd International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

2. Preliminaries

We address the challenge of sampling from a target probability distribution on $\mathbb{R}^d$ of the form $\pi(\mathbf{x}) = \exp(-\mathcal{E}(\mathbf{x}))/Z$ where $Z = \int_{\mathbb{R}^d} \exp(-\mathcal{E}(\mathbf{x}))d\mathbf{x}$ is unknown but the energy function $\mathcal{E}(\mathbf{x})$ is known pointwise. Classical approaches to this problem rely on Monte Carlo methods, including importance sampling (IS), sequential Monte Carlo (SMC; Del Moral et al., 2006), and Markov chain Monte Carlo (MCMC; Hastings, 1970; Kass et al., 1998), along with their various extensions. More recently, advances in deep generative models have spurred efforts to train amortized samplers — parameterized models capable of directly generating samples from the target distribution π (Vargas et al., 2022; Zhang & Chen, 2022; Vargas et al., 2023, see §4).

Diffusion samplers. Diffusion models assume a generative process governed by a stochastic differential equation (SDE):

$$d\mathbf{x}_t = u_\theta(\mathbf{x}_t, t)dt + \sigma(t)d\mathbf{w}_t, \quad t \in [0, 1], \tag{1}$$

where $\mathbf{x}_0 \sim \mu_0$ and $\mathbf{w}_t$ is standard Brownian motion. The goal is to learn the drift u_θ such that the marginal distribution of $\mathbf{x}_1$ closely approximates the target π.

To implement this, we discretize the SDE via *Euler-Maruyama* scheme (Maruyama, 1955) with predefined time points $0 = t_0 < t_1 < \cdots < t_N = 1$. This yields a discrete-time Markov process with transition kernel:

$$\begin{aligned} &q_\theta(\mathbf{x}_{t_{n+1}}|\mathbf{x}_{t_n}) \\ &= \mathcal{N}(\mathbf{x}_{t_{n+1}}; \mathbf{x}_{t_n} + u_\theta(\mathbf{x}_{t_n}, t_n)\Delta t_n, \sigma^2(t_n)\Delta t_n \mathbf{I}_d), \end{aligned} \tag{2}$$

where $\Delta t_n = t_{n+1} - t_n$. The objective of amortized sampling is to ensure that the marginal distribution satisfies:

$$q_\theta^\top(\mathbf{x}_1) = \int q_\theta(\mathbf{x}_{t_{0:N}})d\mathbf{x}_{t_{0:N-1}} \propto \exp(-\mathcal{E}(\mathbf{x}_1)), \tag{3}$$

where $q_\theta(\mathbf{x}_{t_{0:N}}) = q_\theta(\mathbf{x}_{t_0}) \prod_{n=1}^N q_\theta(\mathbf{x}_{t_n}|\mathbf{x}_{t_{n-1}})$ and $q_\theta(\mathbf{x}_{t_0}) = \mu_0(\mathbf{x}_0)$ (also recall that $t_0 = 0$ and $t_N = 1$).

Trajectory balance. If the support of the joint distribution conditioned on a given $\mathbf{x}_1$ is intractable to integrate over, the constraint (3) cannot be enforced directly. Instead, we can introduce auxiliary backward kernels $p_\phi(\mathbf{x}_{t_{n-1}}|\mathbf{x}_{t_n})$ and try to minimize a divergence between the forward and backward joint distributions over the trajectories, *e.g.*, the reverse Kullback–Leibler (KL) divergence $D_{\mathrm{KL}}(q_\theta(\mathbf{x}_{t_{0:N}})\|\pi(\mathbf{x}_{t_N}) \cdot p_\phi(\mathbf{x}_{t_{0:N-1}}|\mathbf{x}_{t_N}))$. Note that, by the data processing inequality, the trajectory-level divergence upper-bounds the divergence between the marginals.

Trajectory balance (TB; Malkin et al., 2022) is an off-policy objective for minimizing such trajectory-level divergences. For a given trajectory $\mathbf{x}_{t_{0:N}}$, the TB loss is defined as

$$\mathcal{L}_{\mathrm{TB}}(\mathbf{x}_{t_{0:N}}) = \left(\log \frac{Z_\theta q_\theta(\mathbf{x}_{t_{0:N}})}{\exp(-\mathcal{E}(\mathbf{x}_1))p_\phi(\mathbf{x}_{t_{0:N-1}}|\mathbf{x}_{t_N})}\right)^2. \tag{4}$$

Here, Z_θ is a learnable parameter that approximates the normalizing constant Z. When this loss is minimized over all possible trajectories, we achieve $q_\theta^\top = \pi$ and $Z_\theta = Z$. To lighten the notation, we denote trajectories $\mathbf{x}_{t_{0:N}}$ as τ in subsequent sections, unless it causes any confusion.

Importance sampling. Consider computing the expectation of an integrable function $h(\mathbf{y})$ under an unknown target distribution $p(\mathbf{y}) = \tilde{p}(\mathbf{y})/Z_\mathcal{Y}$ where $\mathbf{y} \in \mathbb{R}^d$ and $Z_\mathcal{Y} = \int_{\mathbb{R}^d} \tilde{p}(\mathbf{y})d\mathbf{y}$:

$$I = \mathbb{E}_p[h(\mathbf{y})] = \int_{\mathbb{R}^d} h(\mathbf{y})p(\mathbf{y})d\mathbf{y}. \tag{5}$$

Since drawing independent samples from $p(\mathbf{y})$ is intractable, standard Monte Carlo estimation is not applicable. Importance sampling circumvents this by introducing a tractable proposal distribution $q(\mathbf{y})$ such that $q(\mathbf{y}) > 0$ whenever $h(\mathbf{y})p(\mathbf{y}) \neq 0$, and rewriting the integral as:

$$I = \mathbb{E}_q[w(\mathbf{y})h(\mathbf{y})] = \frac{1}{Z_\mathcal{Y}} \int_{\mathbb{R}^d} w(\mathbf{y})h(\mathbf{y})q(\mathbf{y})d\mathbf{y}, \tag{6}$$

where $w(\mathbf{y}) = \tilde{p}(\mathbf{y})/q(\mathbf{y})$ is the importance weight. When $Z_\mathcal{Y}$ is known, importance sampling gives an unbiased estimator $\bar{I}_{IS} = \frac{1}{KZ}\sum_{k=1}^K w(\mathbf{y}^k)h(\mathbf{y}^k)$ for $\mathbf{y}^k \overset{\text{i.i.d.}}{\sim} q(\mathbf{y})$. However, since $Z_\mathcal{Y}$ is typically unknown, we use the self-normalized importance sampling estimator:

$$\widehat{I}_{IS} = \frac{1}{K}\sum_{k=1}^K W^k h(\mathbf{y}^k), \tag{7}$$

where $W^k = \frac{w(\mathbf{y}^k)}{\sum_{j=1}^K w(\mathbf{y}^j)}$ are the self-normalized importance weights. This estimator is consistent, meaning $\widehat{I}_{IS} \to I$ as $K \to \infty$.

More interestingly, we can also approximate the target distribution p using the importance-weighted empirical measure:

$$\widehat{p}(\mathbf{y}; \mathbf{y}^{1:K}) = \sum_{k=1}^K W^k \delta_{\mathbf{y}^k}(\mathbf{y}), \tag{8}$$

where $\delta_{\mathbf{y}^k}(\mathbf{y})$ is the delta measure at $\mathbf{y}^k$. This empirical measure $\widehat{p}(\mathbf{y}; \mathbf{y}^{1:K})$ converges weakly to the true target distribution $p(\mathbf{y})$ as $K \to \infty$, which implies that for any bounded, continuous function $g(\mathbf{y})$, $\mathbb{E}_{\widehat{p}}[g(\mathbf{y})] = \sum_{k=1}^K W^k g(\mathbf{y}) \to \mathbb{E}_p[g(\mathbf{y})]$ as $K \to \infty$.

3. Methods

We explore how importance sampling can improve the training of the diffusion samplers, especially when using the Trajectory Balance (TB) objective (4).

Since TB is an off-policy objective, we can train our sampler using trajectories drawn from any **training (behavior) distribution** $\eta(\tau)$ with full support. The choice of η plays a crucial role in both training efficiency and the quality of the resulting sampler q_θ, especially when the target distribution is multi-modal, where η should ideally explore the regions that our sampler q_θ has low density (Kim et al., 2025a; Pan et al., 2023; Phillips & Cipcigan, 2024).

Throughout this and the subsequent sections, we assume that there exists a **desired training distribution** η_* that we want to train our sampler on. While η_* may not be tractable to sample from directly, we require that its unnormalized density $\widetilde{\eta}_*$ can be evaluated pointwise, *e.g.*, $\eta_* = \pi \cdot p_\phi$ (see §3.4 for discussions about the choices of η_*). Our goal is to approximate the expected gradient with respect to θ under this distribution:

$$G_{\eta_*} = \mathbb{E}_{\eta_*}[\nabla_\theta \mathcal{L}_{\text{TB}}(\tau)], \tag{9}$$

using importance sampling and its variants.

3.1. Importance-weighted Training

Consider another **proposal distribution** $\eta_\beta(\tau)$ that we can easily sample from and also has full support (*e.g.*, q_θ). Following (8), we approximate the desired training distribution η_* using samples from η_β:

$$\widehat{\eta}(\tau; \tau^{1:K}) = \sum_{k=1}^{K} W^k \delta_{\tau^k}(\tau), \tag{10}$$

where $\tau^k \overset{\text{i.i.d.}}{\sim} \eta_\beta$ and the importance weights are:

$$W^k = \frac{w^k}{\sum_{j=1}^K w^j}, \quad w^k = w(\tau^k) = \frac{\widetilde{\eta}_*(\tau^k)}{\eta_\beta(\tau^k)}. \tag{11}$$

Here, $\widetilde{\eta}_*$ is the unnormalized density of η_* that we can evaluate. Also, we obtain a consistent estimate for G_{η_*} in (9):

$$G_{\widehat{\eta}} = \mathbb{E}_{\widehat{\eta}}[\nabla_\theta \mathcal{L}_{\text{TB}}(\tau)] = \sum_{k=1}^{K} W^k \nabla_\theta \mathcal{L}_{\text{TB}}(\tau^k). \tag{12}$$

As with (6) and (8), consistency ensures that $\widehat{\eta} \to \eta_*$ and $G_{\widehat{\eta}} \to G_{\eta_*}$, as $K \to \infty$. This approach leads to Algorithm 1, which presents the basic version of importance-weighted training.

3.2. Importance-weighted Experience Replay

Note that the asymptotic bias and variance of (12) are both $\mathcal{O}(1/K)$ (Owen, 2013, Chapter 9.2). To reduce the variance of this estimator, we can leverage the history of importance-weighted samples collected from past iterations by storing them in a replay buffer.

Let η_β^m be the proposal distribution at m-th training round. Without loss of generality, let the replay buffer $\mathcal{B}$ store the first M batches of trajectories with their corresponding importance weights, *i.e.*, $\mathcal{B} = (\tau^{1:M,1:K}, w^{1:M,1:K})$, where $w^{m,k} = \frac{\widetilde{\eta}_*(\tau^{m,k})}{\eta_\beta^m(\tau^{m,k})}$.

The key challenge is to effectively combine these historical samples, each generated using a different proposal distribution, to obtain better gradient estimates. We present two approaches below.

3.2.1. Uniform mixture over batches

A straightforward approach is to treat the replay buffer as a uniform mixture over batches, applying importance weighting within each batch independently. This approximates the desired training distribution η_* as:

$$\begin{aligned}\widehat{\eta}_{\text{uni}}(\tau; \mathcal{B}) &= \frac{1}{M}\sum_{m=1}^{M}\sum_{k=1}^{K} W^{m,k}\delta_{\tau^{m,k}}(\tau) \\ &= \frac{1}{M}\sum_{m=1}^{M} \widehat{\eta}(\tau; \tau^{m,1:K}),\end{aligned} \tag{13}$$

where $W^{m,k} = \frac{w^{m,k}}{\sum_{j=1}^K w^{m,j}}$ are the normalized weights within batch m, and $\widehat{\eta}$ is defined as in (10). When $K \to \infty$, since each $\widehat{\eta}(\tau; \tau^{m,1:K})$ converges to η_* as $K \to \infty$, their uniform mixture $\widehat{\eta}_{\text{uni}}$ also converges to η_*.

The corresponding gradient estimator for G_{η_*} in (9) then becomes:

$$\begin{aligned}G_{\widehat{\eta}_{\text{uni}}} &= \mathbb{E}_{\widehat{\eta}_{\text{uni}}}[\nabla_\theta \mathcal{L}_{\text{TB}}(\tau^{m,k})] \\ &= \frac{1}{M}\sum_{m=1}^{M}\sum_{k=1}^{K} W^{m,k}\nabla_\theta \mathcal{L}_{\text{TB}}(\tau^{m,k}) \\ &\approx \frac{1}{K}\sum_{k=1}^{K} \nabla_\theta \mathcal{L}_{\text{TB}}(\tau^k),\end{aligned} \tag{14}$$

where the last line is the Monte Carlo approximation with samples $\tau^k \overset{\text{i.i.d.}}{\sim} \widehat{\eta}_{\text{uni}}(\tau; \mathcal{B})$. While the exact estimate (second line) is available, it requires M times more memory than typical batch processing, so we use the approximation in practice.

3.2.2. Pooled importance weighting

An alternative approach, which we refer to as "pooled importance weighting," directly combines all $M \times K$ samples

by pooling their original unnormalized weights $w^{m,k}$ and then normalizing them globally across the entire buffer. This creates a single, comprehensive empirical distribution from all historical data. The resulting approximation of η_* is:

$$\begin{aligned}\widehat{\eta}_{\text{pool}}(\tau;\mathcal{B}) &= \frac{1}{\sum_{l=1}^{M}\sum_{j=1}^{K} w^{l,j}} \sum_{m=1}^{M}\sum_{k=1}^{K} w^{m,k}\delta_{\tau^{m,k}}(\tau) \\ &= \sum_{m=1}^{M} \overline{W}^m \widehat{\eta}(\tau;\tau^{m,1:K}),\end{aligned} \tag{15}$$

where $\overline{W}^m = \frac{\widehat{Z}^m}{\sum_{l=1}^{M}\widehat{Z}^l}$, $\widehat{Z}^m = \frac{1}{K}\sum_{k=1}^{K} w^{m,k}$ is the estimate of the normalizing constant from m-th batch, and $\widehat{\eta}$ is defined as in (10). Note that $\widehat{\eta}_{\text{pool}}$ is a weighted mixture of the individual batch-specific empirical distributions, with mixture weights $\overline{W}^m$.

An advantage of $\widehat{\eta}_{\text{pool}}$ over $\widehat{\eta}_{\text{uni}}$ in (13) is that $\widehat{\eta}_{\text{pool}}$ approaches η_* as the total number of samples $M \times K \to \infty$, potentially offering a more robust approximation. This pooling approach aligns with Group Importance Sampling (see Appendix B of Martino et al. (2018) for details).

As with the uniform mixture approach in (14), we obtain gradient estimates for G_{η_*} in (9) by $G_{\widehat{\eta}_{\text{pool}}} = \mathbb{E}_{\widehat{\eta}_{\text{pool}}}[\nabla_\theta \mathcal{L}_{\text{TB}}(\tau)]$, which we approximate using samples from $\widehat{\eta}_{\text{pool}}$.

Both the uniform mixture (§3.2.1) and the pooled importance weighting (§3.2.2) require sampling from their respective empirical distributions ($\widehat{\eta}_{\text{uni}}$ and $\widehat{\eta}_{\text{pool}}$, respectively). This sampling step can be efficiently implemented using prioritized experience replay (Schaul et al., 2016), using the corresponding importance weights as the priority scores. See Algorithm 2 for the resulting algorithms.

3.3. Adaptive Importance Weight Tempering

When our proposal η_β differs significantly from the desired training distribution η_*, the importance weights can exhibit extremely high variance. This becomes particularly severe when η_* involves a target distribution π with sharp, well-separated modes. High variance in the importance weights leads to the weight degeneracy problem: after normalization, only a few samples receive significant weights while the rest become negligible. Such degeneracy significantly increases the variance of importance-weighted gradient estimates, *e.g.*, (12), causing instability in the training.

To address this issue, we temper the importance weights using an inverse temperature parameter $\lambda \in [0,1]$, transforming $w \mapsto w^\lambda$ before normalization. This is equivalent to replacing the desired training distribution η_* with $\eta_*(\tau)^\lambda \eta_\beta(\tau)^{1-\lambda}$, a geometric interpolation of η_* and η_β, since:

$$w^\lambda = \left(\frac{\eta_*(\tau)}{\eta_\beta(\tau)}\right)^\lambda = \frac{\eta_*(\tau)^\lambda \eta_\beta(\tau)^{1-\lambda}}{\eta_\beta(\tau)}. \tag{16}$$

While this could significantly reduce variance, it also introduces bias to our approximations. Thus, rather than using a fixed λ, we adaptively set λ to maintain a minimum level of sample diversity, measured by the Effective Sample Size (ESS). ESS represents the ratio of variances between plain Monte Carlo and self-normalized importance sampling estimators. Since we cannot compute the plain Monte Carlo variance directly, ESS is commonly approximated as:

$$\widehat{\text{ESS}}(w^{1:K}) = \frac{\left(\sum_{k=1}^{K} w^k\right)^2}{\sum_{k=1}^{K}\left(w^k\right)^2}, \tag{17}$$

originally proposed by Kong (1992) and recently studied by Elvira et al. (2022). This approximation is widely used in adaptive resampling schemes for sequential Monte Carlo. Note that the value $\widehat{\text{ESS}}$ ranges from 1 (complete degeneracy) to K (uniform weights).

Our adaptive tempering scheme selects the largest λ that maintains $\widehat{\text{ESS}}$ above a specified threshold:

$$\lambda^* = \max\left\{\lambda \in [0,1] : \widehat{\text{ESS}}\left(\left(w^\lambda\right)^{1:K}\right) \geq \gamma K\right\}, \tag{18}$$

where $\gamma \in [0,1]$ is a user-defined threshold. Since $\widehat{\text{ESS}}\left(\left(w^\lambda\right)^{1:K}\right)$ decreases monotonically from K (when $\lambda = 0$) to $\widehat{\text{ESS}}(w^{1:K})$ (when $\lambda = 1$), we can efficiently find a fairly good approximation of λ^* using binary search with negligible computational overhead.

In our experiments, we set $\eta_* = \pi \cdot p_\phi$ and $\eta_\beta = q_\theta$. As training progresses and q_θ better approximates the target, we expect λ^* to gradually increase toward 1, since the model learns to minimize importance weight variance (see §3.4).

3.4. Practical Considerations

Choice of η_* and η_β. Two key design decisions are selecting the desired training distribution η_* and the tractable proposal distribution η_β. The most straightforward choice for η_* would be $\eta_*(\tau) = \pi(\mathbf{x}_1) \cdot p_\phi(\tau|\mathbf{x}_1)$ using a fixed backward policy p_ϕ, which we adopt in our experiments. Alternative formulations are possible, such as using a tempered target $\pi^\lambda \cdot p_\phi$ (with $\lambda \in \mathbb{R}_+$) or incorporating loss-dependent terms as in Kim et al. (2025a).

For the proposal distribution, we use the current policy q_θ being trained, *i.e.*, $\eta_\beta^m = q_\theta^m$ at training round m. A simple alternative is to use a variant of q_θ^m by increasing variance in the transition kernels (2), as employed in several previous works (Lahlou et al., 2023; Sendera et al., 2024).

Notably, when we set $\eta_* = \pi \cdot p_\phi$ and $\eta_\beta = q_\theta$, minimization of the TB loss (4) yields the optimal proposal distribution that minimizes the variance of the importance weights, since $w(\tau) = \frac{\widetilde{\eta}_*(\tau)}{\eta_\beta(\tau)} = \frac{\exp(-\mathcal{E}(\mathbf{x}_1))p_\phi(\tau|\mathbf{x}_1)}{q_\theta(\tau)} = Z_\theta$.

Terminal-state buffer. The approach described in §3.2 requires storing complete trajectories τ in the replay buffer, which is memory-intensive and limits the number of trajectories we can retain. However, when our desired training distribution takes the form $\eta_*(\tau) = \eta_*^\top(\mathbf{x}_1)p_\phi(\tau|\mathbf{x}_1)$ with a fixed backward policy p_ϕ, we can significantly reduce memory consumption by storing only the terminal states $\mathbf{x}_1$.

The key insight is that if weighted trajectories follow the distribution η_*, then the corresponding weighted terminal states automatically follow the marginal distribution $\eta_*^\top$. This allows us to reuse trajectory-level importance weights to construct an approximation of the terminal-state distribution. For instance, corresponding to the trajectory-level uniform mixture in (13), we define the terminal-state approximation:

$$\widehat{\eta}^\top_{\text{uni}}(\mathbf{x}_1; \mathcal{B}_\mathcal{X}) = \frac{1}{M}\sum_{m=1}^{M}\sum_{k=1}^{K} W^{m,k}\delta_{\mathbf{x}_1^{m,k}}(\mathbf{x}_1), \qquad (19)$$

where $\mathcal{B}_\mathcal{X}$ contains only terminal states and $W^{m,k}$ are the same importance weights used in (13).

For training, we sample terminal states from this importance-weighted terminal-state buffer, and then generate complete trajectories by sampling backwards using the fixed policy p_ϕ. This approach dramatically reduces memory requirements while maintaining the same statistical properties as the full trajectory buffer.

4. Related Works

Inspired by the successes of diffusion models in approximating empirical distributions by denoising processes (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021b), diffusion samplers, beginning with Vargas et al. (2022); Zhang & Chen (2022); Vargas et al. (2023), are generative models of the same form that are trained to sample a distribution that can be queried for its energy (and possibly its gradient), but not sampled from directly. The problem of training a diffusion sampler is known to have an interpretation as stochastic optimal control (Zhang & Chen, 2022; Nüsken & Richter, 2021; Berner et al., 2024).

Diffusion samplers can be trained by off-policy methods, that is, ones minimizing some discrepancy between a generative process and its reverse without differentiating through the sampling process. Such methods were independently introduced from the path space measure perspective (Richter et al., 2020; 2024) and that of generative flow networks, general off-policy RL algorithms for sampling from unnormalized densities, in works including Lahlou et al. (2023) (further developed in Zhang et al. (2024); Sendera et al. (2024); Kim et al. (2025a). A unifying perspective on all objectives and continuous-time limit analysis is provided in Berner et al. (2025). Various off-policy behavior policies for diffusion samplers have been proposed: these policies attempt to modify the distribution of states seen during training so as to lead to better coverage of the modes of the target distribution by the trained sampler. Behavior policies using auxiliary Monte Carlo exploration (Sendera et al., 2024) and an additional trained policy guided by the loss of the sampler (Kim et al., 2025a) show promise in this regard. Our simple and principled approach of using Monte Carlo methods for training trajectory selection is inspired by these works and shows further improvements in the training of diffusion samplers.

Several works have explored the connection between diffusion models or samplers and sequential Monte Carlo or importance sampling, albeit in different ways than this paper. For diffusion models, Monte Carlo methods, including SMC, have been proposed in the setting of sampling a posterior distribution under a diffusion prior (Doucet et al., 2022; Cardoso et al., 2024; Song et al., 2023; Dou & Song, 2024). These sampling algorithms run at inference time, rather than guiding the training of an amortized sampler that can provide unbiased samples in finite time at convergence. In Máté & Fleuret (2023), the training objective for a denoizer maintains the sampler in balance with (learned) intermediate target distributions, amounting to an amortized adaptive proposal for twisted SMC in the continuous-time limit; related ideas are explored in Vargas et al. (2024); Chen et al. (2025); Albergo & Vanden-Eijnden (2025). However, these methods are typically coupled to a specialized training objective, while we are interested in behavior policy selection for general off-policy training losses.

Finally, it is worth noting that our proposed algorithms are not specific to diffusion sampler training and are in theory applicable to any hierarchical sampler. Monte Carlo methods have been used in GFlowNet training, with Monte Carlo tree search for forward exploration (Morozov et al., 2024) and both appropriate Monte Carlo exploration in the target space (Zhang et al., 2022; Kim et al., 2024; Sendera et al., 2024; Kim et al., 2025a) and replay buffers for off-policy training (Deleu et al., 2022; Tiapkin et al., 2024; Sendera et al., 2024, among others) showing success in domains more general than diffusion samplers.

5. Experiments

We evaluate our importance-weighted training methods on synthetic target distributions (§5.1) and the Boltzmann distribution of Alanine Dipeptide (§5.2).

We refer to the basic importance-weighted training (§3.1) as "**IW-Training**," importance-weighted experience replay with uniform mixture approach (§3.2.1) as "**UIW-Buffer**," and pooled importance weighting (§3.2.2) as "**PIW-Buffer**."

Table 1. Results on synthetic targets. For each metric, we report the mean and standard deviation from five independent runs. The best mean values and the second-best mean values among all algorithms are highlighted with **bold** and underline, respectively.

Target →	GMM40 ($d = 2$)				ManyWell ($d = 32$)			
Algorithm ↓ Metric →	ELBO (↑)	IW-ELBO (↑)	EUBO (↓)	Sinkhorn (↓)	ELBO (↑)	IW-ELBO (↑)	EUBO (↓)	Sinkhorn (↓)
PIS	-2.97 ± 0.32	-2.46 ± 0.27	267.81 ± 19.94	827.34 ± 299.01	161.03 ± 0.01	161.99 ± 0.02	299.57 ± 18.41	35.28 ± 0.04
TB	-3.13 ± 0.01	-2.60 ± 0.00	273.00 ± 19.96	680.47 ± 9.69	160.96 ± 0.01	161.99 ± 0.02	315.72 ± 8.34	35.28 ± 0.02
+ ϵ-expl.	-1.73 ± 0.06	-1.20 ± 0.07	143.19 ± 24.74	439.56 ± 8.98	160.99 ± 0.01	161.99 ± 0.02	341.34 ± 10.45	35.28 ± 0.02
+ IW-Training (ours)	-1.18 ± 0.03	-0.35 ± 0.01	68.56 ± 8.51	223.33 ± 0.91	160.90 ± 0.01	161.99 ± 0.02	316.62 ± 10.09	35.35 ± 0.03
+ Buffer	-2.34 ± 0.42	-1.67 ± 0.33	109.25 ± 64.42	506.05 ± 51.80	160.97 ± 0.03	162.31 ± 0.41	212.08 ± 5.90	35.06 ± 0.24
+ R-Buffer	-1.83 ± 0.08	-1.17 ± 0.21	34.52 ± 7.76	372.93 ± 91.60	161.61 ± 0.12	164.37 ± 0.98	174.75 ± 1.60	30.80 ± 0.78
+ L-Buffer	-1.85 ± 0.28	-0.89 ± 0.13	112.19 ± 56.39	312.93 ± 36.38	161.75 ± 0.29	163.97 ± 0.22	176.92 ± 4.77	26.82 ± 1.44
+ UIW-Buffer (ours)	**-1.03** ± 0.03	**0.01** ± 0.01	**0.53** ± 0.01	**1.01** ± 0.03	163.04 ± 0.04	164.69 ± 0.02	166.04 ± 0.02	23.16 ± 0.05
+ PIW-Buffer (ours)	-1.09 ± 0.02	**0.01** ± 0.02	0.55 ± 0.00	1.08 ± 0.02	**163.07** ± 0.06	**164.71** ± 0.02	**166.03** ± 0.03	**23.03** ± 0.04

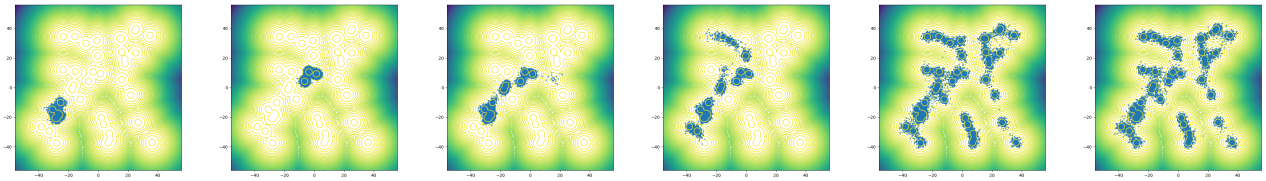

Figure 1. Samples obtained from samplers trained with each algorithm on GMM40 for the first random seed. The proposed importance-weighted scheme (UIW- and PIW-Buffer) significantly improves the mode coverage.

We compare our methods against several existing off-policy training schemes for Trajectory Balance (TB), including ϵ-exploration (ϵ-expl.; Lahlou et al., 2023), a naive buffer without prioritization, a reward-prioritized buffer (R-Buffer; Shen et al., 2023), and a loss-prioritized buffer (L-Buffer; Schaul et al., 2016). We also include the path-integral sampler (PIS; Zhang & Chen, 2022), which uses the same variance-exploding diffusion process as our TB variants. For all algorithms, we excluded the Langevin preconditioning scheme introduced in Zhang & Chen (2022) and recently analyzed by He et al. (2025), since it significantly increases the number of energy evaluations and also makes training highly unstable in the Alanine Dipeptide task.

5.1. Synthetic Targets

We consider two challenging synthetic target distributions with multiple well-separated modes: a 2-dimensional mixture of Gaussians with 40 mixture components (GMM40) from (Midgley et al., 2023), and a 32-dimensional ManyWell distribution, constructed as the product of 16 identical 2-dimensional double-well potentials (Noé et al., 2019; Nüsken & Richter, 2021).

The multi-modality of these targets poses a fundamental exploration challenge where a sampler needs to be trained to closely approximate the target while simultaneously exploring the space to discover the modes. Methods that minimize the reverse KL divergence, including PIS and on-policy TB, typically struggle with mode collapse on these distributions due to the mode-seeking nature of reverse KL divergence (Minka et al., 2005). The off-policy learning objectives, such as TB, offer a distinct advantage in terms of mode-coverage, allowing for stable training with exploratory off-policy samples collected from different policies (Malkin et al., 2023).

We evaluate using four metrics: evidence lower bound (ELBO), evidence upper bound (EUBO; Blessing et al., 2024), importance-weighted ELBO (IW-ELBO), and Sinkhorn distance (Cuturi, 2013). Please refer to Appendix B.1 for detailed target specifications and hyperparameter settings, and Appendix C for metric definitions.

Results. The results are summarized in Table 1, with Fig. 1 showing samples from samplers trained with each algorithm on GMM40. From high EUBO values, we can see, PIS and on-policy TB suffer severe mode collapse as expected, covering only a subset of target modes. IW-Training shows improved performance over on-policy TB on GMM40 but not on ManyWell, likely due to increased gradient variance from importance weighting in higher dimensions. In contrast, both UIW- and PIW-Buffer demonstrate remarkable performance, outperforming other buffer prioritization schemes by substantial margins across both tasks. These results suggest significant potential for advancing off-policy training of amortized samplers (Lahlou et al., 2023; Sendera et al., 2024; Richter et al., 2024), where replay buffers are widely adopted but typically rely on naive implementations or heuristic prioritization.

Table 2. Results of distribution-based metrics on molecular conformer generation in Alanine Dipeptide. For each metric, we report the mean and standard deviation obtained from five independent runs. "×" indicates that training failed due to numerical error.

Algorithm ↓ Metric →	ELBO (↑)	IW-ELBO (↑)	EUBO (↓)
PIS	×	×	×
TB	×	×	×
+ Buffer	-755.9 ± 268.3	-274.1 ± 35.3	-125.8 ± 1.4
+ R-Buffer	-797.7 ± 441.2	-184.9 ± 11.5	-149.4 ± 5.7
+ L-Buffer	-494.3 ± 47.0	-303.0 ± 33.7	-112.2 ± 5.5
+ UIW-Buffer (ours)	-217.1 ± 18.1	-173.9 ± 0.5	-159.5 ± 3.6
+ PIW-Buffer (ours)	-199.9 ± 2.7	-173.7 ± 0.6	-158.6 ± 3.5

5.2. Alanine Dipeptide

We evaluate our methods for approximating the Boltzmann distribution of alanine dipeptide (ALDP), a 22-atom molecule in an implicit solvent at a temperature of 300K. This represents a challenging real-world molecular sampling task that has become a standard benchmark for Boltzmann generators (Noé et al., 2019; Wu et al., 2020).

ALDP presents several key challenges: the energy landscape contains multiple well-separated modes with high barriers between them, and the high-dimensional coordinate space (66 Cartesian coordinates) requires careful preprocessing. Following Midgley et al. (2023), we employ internal coordinates and apply transformations relative to the minimum energy configuration, thereby reducing the dimensionality from 66 to 60.

We employ the same MLP architecture as in §5.1 but with increased capacity to handle the higher complexity of this task. Additionally, we occasionally augment the buffer using MCMC starting from existing buffer samples, given the difficulty of exploring the highly multi-modal energy landscape. See Appendix B.2 for detailed experimental settings, including the details of the MCMC procedure.

We evaluate performance using ELBO, IW-ELBO, EUBO, and Ramachandran plots. For EUBO calculations and reference Ramachandran plots, we use the validation samples provided by Midgley et al. (2023).

Results. Table 2 summarizes the distribution metrics, and Fig. 2 shows the Ramachandran plots drawn with samples drawn from the models trained with each algorithm. Due to the highly peaky energy landscape, algorithms without the buffer failed during the optimization process. The proposed UIW- and PIW-buffer schemes significantly improve the results in terms of both distribution metrics and Ramachandran plots. Still, substantial room for improvement remains, such as using advanced network architectures, finer discretization steps, incorporating likelihood maximization of buffer samples, or employing more powerful MCMC algorithms.

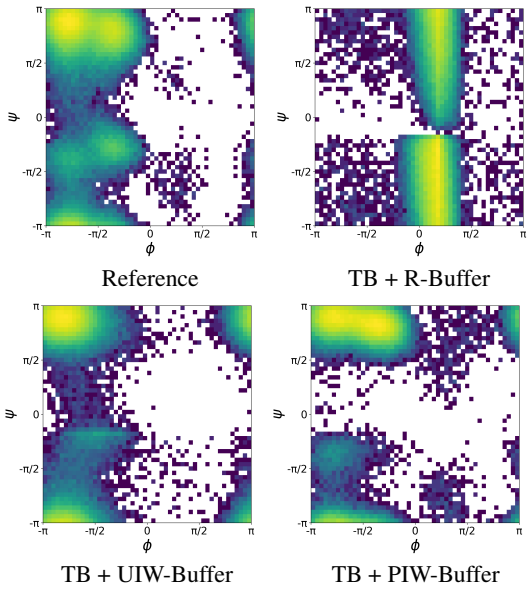

Figure 2. Ramachandran plots generated with 100,000 samples drawn from samplers trained with each algorithm for the first random seed. The reference samples are taken from (Midgley et al., 2023).

6. Conclusion

Contributions. In this work, we introduced several approaches for incorporating importance sampling ideas into the off-policy training of diffusion samplers, with the goal of combining the strengths of amortized sampling and Monte Carlo techniques. Specifically, we developed an importance-weighted training framework to approximate a desired training distribution that provides better learning signals. We also presented practical variants of this approach by leveraging a history of weighted samples stored in a replay buffer. When combined with the proposed stabilization techniques, our methods demonstrated substantial improvements over on-policy training and replay buffers with other prioritization strategies. Finally, we validated our algorithms on molecular conformer generation for the alanine dipeptide system and demonstrated their potential for real-world applications.

Our work contributes to the growing body of research that connects learning-based sampling with Monte Carlo methods (Chen et al., 2025; Albergo & Vanden-Eijnden, 2025). While sharing similarities with existing work, our approach is more general in that our framework does not assume any specific form in training loss, requiring only a general off-policy training loss. Additionally, we highlighted the critical importance of replay training – a widely adopted technique that is often implemented in simplistic ways that do not release its full potential – and showed how it can be significantly enhanced through principled importance sampling strategies.

Future research directions. As discussed in §3.4, we used the target distribution with the backward kernel as our desired training distribution. There may exist a better choice that possibly adapts to indicate the most informative regions in terms of the current sampler, similar to Kim et al. (2025a).

Another promising future direction would be investigating how our importance-weighted experience replay schemes perform with objectives beyond TB, such as subtrajectory balance (Madan et al., 2023). Subtrajectory objectives were studied for diffusion samplers in Zhang et al. (2024) but found in Sendera et al. (2024) to underperform TB and its close relative, log-variance loss (Richter et al., 2020), due to high cost and instability. However, results in the few-step setting (Berner et al., 2025) suggest subtrajectory balance and detailed balance may see their intended credit assignment benefits realized with appropriate training policies such as the IW-based ones we propose. These methods involve learning approximations to intermediate target densities, as in twisted SMC (Lawson et al., 2018; 2022), and can thus be combined with predictor-corrector schemes.

References

Albergo, M. S. and Vanden-Eijnden, E. NETS: A non-equilibrium transport sampler. *International Conference on Machine Learning (ICML)*, 2025.

Berner, J., Richter, L., and Ullrich, K. An optimal control perspective on diffusion-based generative modeling. *Transactions on Machine Learning Research (TMLR)*, 2024.

Berner, J., Richter, L., Sendera, M., Rector-Brooks, J., and Malkin, N. From discrete-time policies to continuous-time diffusion samplers: Asymptotic equivalences and faster training. *arXiv preprint arXiv:2501.06148*, 2025.

Blessing, D., Jia, X., Esslinger, J., Vargas, F., and Neumann, G. Beyond ELBOs: A large-scale evaluation of variational methods for sampling. *International Conference on Machine Learning (ICML)*, 2024.

Burda, Y., Grosse, R. B., and Salakhutdinov, R. Importance weighted autoencoders. *International Conference on Learning Representations (ICLR)*, 2016.

Cardoso, G., el idrissi, Y. J., Corff, S. L., and Moulines, E. Monte carlo guided denoising diffusion models for bayesian linear inverse problems. *International Conference on Learning Representations (ICLR)*, 2024.

Chen, J., Richter, L., Berner, J., Blessing, D., Neumann, G., and Anandkumar, A. Sequential controlled Langevin diffusions. *International Conference on Learning Representations (ICLR)*, 2025.

Cuturi, M. Sinkhorn distances: Lightspeed computation of optimal transport. *Neural Information Processing Systems (NIPS)*, 2013.

Cuturi, M., Meng-Papaxanthos, L., Tian, Y., Bunne, C., Davis, G., and Teboul, O. Optimal transport tools (ott): A jax toolbox for all things wasserstein. *arXiv preprint arXiv:2201.12324*, 2022.

Del Moral, P., Doucet, A., and Jasra, A. Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 68(3):411–436, 2006.

Deleu, T., Góis, A., Emezue, C., Rankawat, M., Lacoste-Julien, S., Bauer, S., and Bengio, Y. Bayesian structure learning with generative flow networks. *Uncertainty in Artificial Intelligence (UAI)*, 2022.

Dou, Z. and Song, Y. Diffusion posterior sampling for linear inverse problem solving: A filtering perspective. *International Conference on Learning Representations (ICLR)*, 2024.

Doucet, A., Grathwohl, W. S., Matthews, A. G., and Strathmann, H. Score-based diffusion meets annealed importance sampling. *Neural Information Processing Systems (NeurIPS)*, 2022.

Duane, S., Kennedy, A., Pendleton, B. J., and Roweth, D. Hybrid Monte Carlo. *Physics Letters B*, 195(2):216–222, 1987.

Eastman, P., Galvelis, R., Peláez, R. P., Abreu, C. R., Farr, S. E., Gallicchio, E., Gorenko, A., Henry, M. M., Hu, F., Huang, J., et al. Openmm 8: molecular dynamics simulation with machine learning potentials. *The Journal of Physical Chemistry B*, 128(1):109–116, 2023.

Elvira, V., Martino, L., and Robert, C. P. Rethinking the effective sample size. *International Statistical Review*, 90 (3):525–550, 2022.

Halton, J. H. Sequential Monte Carlo. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 58, pp. 57–78. Cambridge University Press, 1962.

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

He, J., Du, Y., Vargas, F., Zhang, D., Padhy, S., OuYang, R., Gomes, C., and Hernández-Lobato, J. M. No trick, no treat: Pursuits and challenges towards simulation-free training of neural samplers. In *Workshop on Frontiers in Probabilistic Inference: Sampling Meets Learning @ ICLR 2025*, 2025.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Neural Information Processing Systems (NeurIPS)*, 2020.

Hoffman, M. D., Gelman, A., et al. The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research (JMLR)*, 15(1):1593–1623, 2014.

Holdijk, L., Du, Y., Hooft, F., Jaini, P., Ensing, B., and Welling, M. Stochastic optimal control for collective variable free sampling of molecular transition paths. *Neural Information Processing Systems (NeurIPS)*, 2023.

Kass, R. E., Carlin, B. P., Gelman, A., and Neal, R. M. Markov chain monte carlo in practice: a roundtable discussion. *The American Statistician*, 52(2):93–100, 1998.

Kim, M., Yun, T., Bengio, E., Zhang, D., Bengio, Y., Ahn, S., and Park, J. Local search GFlowNets. *International Conference on Learning Representations (ICLR)*, 2024.

Kim, M., Choi, S., Yun, T., Bengio, E., Feng, L., Rector-Brooks, J., Ahn, S., Park, J., Malkin, N., and Bengio, Y. Adaptive teachers for amortized samplers. *International Conference on Learning Representations (ICLR)*, 2025a.

Kim, M., Seong, K., Woo, D., Ahn, S., and Kim, M. On scalable and efficient training of diffusion samplers. *arXiv preprint arXiv:2505.19552*, 2025b.

Kitagawa, G. Monte carlo filter and smoother for non-gaussian nonlinear state space models. *Journal of computational and graphical statistics*, 5(1):1–25, 1996.

Kong, A. A note on importance sampling using standardized weights. *University of Chicago, Dept. of Statistics, Tech. Rep*, 348:14, 1992.

Lahlou, S., Deleu, T., Lemos, P., Zhang, D., Volokhova, A., Hernández-Garcıa, A., Ezzine, L. N., Bengio, Y., and Malkin, N. A theory of continuous generative flow networks. *International Conference on Machine Learning (ICML)*, 2023.

Lawson, D., Tucker, G., Naesseth, C., Maddison, C., Adams, R., and Teh, Y. Twisted variational sequential Monte Carlo, 2018. URL https://www.cs.utoronto.ca/~cmaddis/pubs/tvsmc.pdf.

Lawson, D., Raventós, A., Warrington, A., and Linderman, S. SIXO: Smoothing inference with twisted objectives. *Neural Information Processing Systems (NeurIPS)*, 2022.

Leimkuhler, B. and Matthews, C. Rational construction of stochastic numerical methods for molecular sampling. *Applied Mathematics Research eXpress*, 2013(1):34–56, 2013.

Leimkuhler, B. J., Matthews, C., and Tretyakov, M. V. On the long-time integration of stochastic gradient systems. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 470, 2014.

Madan, K., Rector-Brooks, J., Korablyov, M., Bengio, E., Jain, M., Nica, A., Bosc, T., Bengio, Y., and Malkin, N. Learning GFlowNets from partial episodes for improved convergence and stability. *International Conference on Machine Learning (ICML)*, 2023.

Malkin, N., Jain, M., Bengio, E., Sun, C., and Bengio, Y. Trajectory balance: Improved credit assignment in GFlowNets. *Neural Information Processing Systems (NeurIPS)*, 2022.

Malkin, N., Lahlou, S., Deleu, T., Ji, X., Hu, E., Everett, K., Zhang, D., and Bengio, Y. GFlowNets and variational inference. *International Conference on Learning Representations (ICLR)*, 2023.

Martino, L., Elvira, V., and Camps-Valls, G. Group importance sampling for particle filtering and mcmc. *Digital Signal Processing*, 82:133–151, 2018.

Maruyama, G. Continuous markov processes and stochastic equations. *Rendiconti del Circolo Matematico di Palermo*, 4:48–90, 1955.

Máté, B. and Fleuret, F. Learning interpolations between Boltzmann densities. *Transactions on Machine Learning Research (TMLR)*, 2023.

Midgley, L. I., Stimper, V., Simm, G. N., Schölkopf, B., and Hernández-Lobato, J. M. Flow annealed importance sampling bootstrap. *International Conference on Learning Representations (ICLR)*, 2023.

Minka, T. et al. Divergence measures and message passing. 2005.

Morozov, N., Tiapkin, D., Samsonov, S., Naumov, A., and Vetrov, D. Improving GFlowNets with Monte Carlo tree search. *arXiv preprint arXiv:2406.13655*, 2024.

Noé, F., Olsson, S., Köhler, J., and Wu, H. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science*, 365(6457):eaaw1147, 2019.

Nüsken, N. and Richter, L. Solving high-dimensional Hamilton–Jacobi–Bellman PDEs using neural networks: perspectives from the theory of controlled diffusions and measures on path space. *Partial Differential Equations and Applications*, 2(4):48, 2021.

Owen, A. B. *Monte Carlo theory, methods and examples*. https://artowen.su.domains/mc/, 2013.

Pan, L., Zhang, D., Courville, A., Huang, L., and Bengio, Y. Generative augmented flow networks. *International Conference on Learning Representations (ICLR)*, 2023.

Phillips, D. and Cipcigan, F. MetaGFN: Exploring distant modes with adapted metadynamics for continuous GFlowNets. *arXiv preprint arXiv:2408.15905*, 2024.

Richter, L., Boustati, A., Nüsken, N., Ruiz, F. J. R., and Ömer Deniz Akyildiz. VarGrad: A low-variance gradient estimator for variational inference. *Neural Information Processing Systems (NeurIPS)*, 2020.

Richter, L., Berner, J., and Liu, G.-H. Improved sampling via learned diffusions. *International Conference on Learning Representations (ICLR)*, 2024.

Schaul, T., Quan, J., Antonoglou, I., and Silver, D. Prioritized experience replay. *International Conference on Learning Representations (ICLR)*, 2016.

Sendera, M., Kim, M., Mittal, S., Lemos, P., Scimeca, L., Rector-Brooks, J., Adam, A., Bengio, Y., and Malkin, N. Improved off-policy training of diffusion samplers. *Neural Information Processing Systems (NeurIPS)*, 2024.

Shen, M. W., Bengio, E., Hajiramezanali, E., Loukas, A., Cho, K., and Biancalani, T. Towards understanding and improving GFlowNet training. *International Conference on Machine Learning (ICML)*, 2023.

Sohl-Dickstein, J., Weiss, E. A., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. *International Conference on Machine Learning (ICML)*, 2015.

Song, J., Zhang, Q., Yin, H., Mardani, M., Liu, M.-Y., Kautz, J., Chen, Y., and Vahdat, A. Loss-guided diffusion models for plug-and-play controllable generation. *International Conference on Machine Learning (ICML)*, 2023.

Song, Y., Durkan, C., Murray, I., and Ermon, S. Maximum likelihood training of score-based diffusion models. *Neural Information Processing Systems (NeurIPS)*, 2021a.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. *International Conference on Learning Representations (ICLR)*, 2021b.

Tiapkin, D., Morozov, N., Naumov, A., and Vetrov, D. Generative flow networks as entropy-regularized RL. *Artificial Intelligence and Statistics (AISTATS)*, 2024.

Vargas, F., Ovsianas, A., Fernandes, D., Girolami, M., Lawrence, N. D., and Nüsken, N. Bayesian learning via neural schrödinger–föllmer flows. *Statistics and Computing*, 33(1):3, 2022.

Vargas, F., Grathwohl, W., and Doucet, A. Denoising diffusion samplers. *International Conference on Learning Representations (ICLR)*, 2023.

Vargas, F., Padhy, S., Blessing, D., and Nüsken, N. Transport meets variational inference: Controlled Monte Carlo diffusions. *International Conference on Learning Representations (ICLR)*, 2024.

Wu, H., Köhler, J., and Noé, F. Stochastic normalizing flows. *Neural Information Processing Systems (NeurIPS)*, 2020.

Zhang, D., Malkin, N., Liu, Z., Volokhova, A., Courville, A., and Bengio, Y. Generative flow networks for discrete probabilistic modeling. *International Conference on Machine Learning (ICML)*, 2022.

Zhang, D., Chen, R. T. Q., Liu, C.-H., Courville, A., and Bengio, Y. Diffusion generative flow samplers: Improving learning signals through partial trajectory optimization. *International Conference on Learning Representations (ICLR)*, 2024.

Zhang, Q. and Chen, Y. Path integral sampler: a stochastic control approach for sampling. *International Conference on Learning Representations (ICLR)*, 2022.

A. Algorithms

Algorithm 1 Importance-weighted Training

Require: q_θ, $\widetilde{\eta}_*$, N_{epoch}, K.
for $m = 1, \dots, N_{\text{epoch}}$ **do**
  Define η_β^m (probably depending on q_θ and m).
  Draw a set of samples $\{\tau^k\}_{k=1}^K$ from η_β^m.
  Calculate the TB loss $\mathcal{L}_{\text{TB}}(\tau^k)$ for each τ^k. {Eq. (4)}
  if DoWeighting(m) **then**
    Calculate the self-normalized importance weight W^k for each τ^k. {Eq. (11)}
  else
    Set $W^k = 1/K$ for all k.
  end if
  Update θ using the importance-weighted gradient: $\theta \leftarrow \text{Optimizer}(\theta, G_{\widehat{\eta}})$. {Eq. (12)}
end for

Algorithm 2 Training with the Importance-weighted Experience Replay

Require: q_θ, $\widetilde{\eta}_*$, N_{epoch}, K, $\mathcal{B}$, IWMethod.
for $m = 1, \dots, N_{\text{epoch}}$ **do**
  if DoBufferSampling(m) **then**
    Draw a set of samples $\{\tau^k\}_{k=1}^K$ from $\mathcal{B}$ by sampling with replacement,
    according to the probability $P(\tau; \mathcal{B}) = \begin{cases} \widehat{\eta}_{\text{uni}}(\tau; \mathcal{B}) & \text{if IWMethod is Uniform } \{\text{Eq. (13)}\} \\ \widehat{\eta}_{\text{pool}}(\tau; \mathcal{B}) & \text{if IWMethod is Pooled. } \{\text{Eq. (15)}\} \end{cases}$
  else
    Define η_β^m (probably depending on q_θ and m).
    Draw a set of samples $\{\tau^k\}_{k=1}^K$ from η_β^m.
    Calculate the importance weights w^k for each τ^k. {Eq. (11)}
    Add the samples in the buffer, $\mathcal{B} \leftarrow \mathcal{B} \cup \{\tau^k, w^k\}_{k=1}^K$.
  end if
  Calculate the TB loss $\mathcal{L}_{\text{TB}}(\tau^k)$ for each τ^k. {Eq. (4)}
  Update θ using the gradient: $\theta \leftarrow \text{Optimizer}\left(\theta, \frac{1}{K}\sum_{k=1}^K \nabla \mathcal{L}_{\text{TB}}(\tau^k)\right)$.
end for

B. Detailed Experimental Settings

B.1. Synthetic Targets

In this section, we formally define the synthetic target distributions and provide details about the model architecture and the hyperparameters used for these tasks.

Target distributions.

- **GMM40** (d=2) (Midgley et al., 2023) is a 2-dimensional mixture of Gaussians with 40 components. Each component has mean $\boldsymbol{\nu}_i$, where each dimension of $\boldsymbol{\nu}_i$ is randomly sampled from $\mathcal{U}[-40, 40]$, and covariance matrix $\mathbf{I}$ (unit variance). The energy for GMM40 is:

$$\mathcal{E}_{\text{GMM40}}(\mathbf{x}) = -\log\left(\frac{1}{40}\sum_{i=1}^{40}\mathcal{N}(\mathbf{x}; \boldsymbol{\nu}_i, \mathbf{I})\right).$$

  We used the same mean values as Midgley et al. (2023).

- **ManyWell** (d=32) (Nüsken & Richter, 2021; Midgley et al., 2023) is constructed as the product of 16 independent copies of a 2-dimensional Double Well distribution (Noé et al., 2019). The Double Well energy is:

$$\mathcal{E}_{\text{DW}}(x_1, x_2) = x_1^4 - 6x_1^2 - \frac{1}{2}x_1 + \frac{1}{2}x_2^2 + \text{const.}.$$

  The ManyWell energy is then defined as:

$$\mathcal{E}_{\text{MW}}(\mathbf{x}) = \sum_{i=1}^{16} \mathcal{E}_{\text{DW}}(x_{2i-1}, x_{2i}).$$

Model architecture and hyperparameters. For the diffusion process and neural network architecture, we follow the established settings from Sendera et al. (2024), which are based on Zhang & Chen (2022). Specifically, we define the backward process as a Brownian bridge where $\mu_0 = \delta_0$ is a point mass. After discretization, this gives the backward transition kernels:

$$p_\phi(\mathbf{x}_{t_n}|\mathbf{x}_{t_{n+1}}) = \mathcal{N}\left(\mathbf{x}_{t_n}; \frac{t_{n+1} - \Delta t_n}{t_{n+1}}\mathbf{x}_{t_{n+1}}, \frac{t_{n+1} - \Delta t_n}{t_{n+1}}\sigma^2 \Delta t_n \mathbf{I}_d\right), \tag{20}$$

where $\Delta t_n = t_{n+1} - t_n$. The forward kernel follows (2), with $\sigma^2(t_n)$ fixed to the constant σ^2 defined in (20). The drift u_θ is parameterized by a Multi-Layer Perceptron (MLP) with hidden dimension 256, which we increased from the setting in Sendera et al. (2024) to compensate for excluding Langevin preconditioning.

For synthetic tasks, we use 50 discretization steps during training and 100 steps for evaluation. To mitigate potential errors from coarse discretization during training, we employ the equidistant discretization scheme recently proposed in Berner et al. (2025).

The key hyperparameter settings are:

- **Common**: We train our sampler for $N_{\text{epoch}} = 15000$ epochs using a batch size of $K = 300$. We use a learning rate of 0.001 for the policy u_θ and 0.1 for $\log Z_\theta$, decaying both by a factor of 0.2 after $0.5N_{\text{epoch}}$ and $0.8N_{\text{epoch}}$ epochs.
- **Energy-specific**: We set σ in (20) to 10.0 for GMM40 and 1.0 for ManyWell.
- **Algorithm-specific**: For the ϵ-exploration baseline, the behavior policy's variance starts at $2\sigma^2$ and linearly decays to σ^2 over the first half of training. For IW-Weighting, we set ESS_{LB} to 0.05 and alternate importance-weighted training and normal training (*i.e.*, DoWeighting(m) in Algorithm 1 line 5 is $[m \bmod 2 == 0]$). For algorithms that use replay buffers, we set the buffer size to 300000 (1000 times the batch size), with a buffer-to-proposal sample ratio of 2:1, meaning that every three epochs consist of two epochs using buffer samples followed by one epoch using proposal samples (*i.e.*, DoBufferSampling(m) in Algorithm 2 line 3 is $[m \bmod 3 == 0]$). Note that the epochs with buffer samples do not require additional energy calls, since energy evaluations occur only for newly drawn proposal samples. For reward- and loss-prioritized buffers, we use rank-based prioritization following Sendera et al. (2024), while for UIW and PIW-Buffer, we use the systematic sampling scheme (Kitagawa, 1996). For importance-weighted experience replay, we set ESS_{LB} to 0.05 for UIW-Buffer, and 0.2 for PIW-Buffer.

B.2. Alanine Dipeptide

We largely follow the experimental setup of Midgley et al. (2023) for the alanine dipeptide (ALDP) task.

Target distribution. Alanine dipeptide is a 22-atom molecule widely used as a benchmark for molecular conformation sampling. The target distribution is the Boltzmann distribution at temperature $T = 300$K:

$$\pi(\mathbf{x}) \propto \exp\left(-\frac{U(\mathbf{x})}{k_B T}\right), \tag{21}$$

where $U(\mathbf{x})$ is the potential energy, k_B is the Boltzmann constant, and $\mathbf{x}$ represents the molecular conformation. Following Midgley et al. (2023), we compute the energy using OpenMM (Eastman et al., 2023).

Coordinate system. Following Midgley et al. (2023), we use internal coordinates (bond lengths, bond angles, and dihedral angles) instead of Cartesian coordinates to ensure translational and rotational invariance (Noé et al., 2019; Midgley et al., 2023). The molecule has 60 internal coordinates in total. The backbone dihedral angles ϕ and ψ are of particular interest as they characterize the major conformational states of the molecule. Moreover, to focus on the L-form samples, we assign a sufficiently high energy to the D-form samples to suppress their generation, unlike Midgley et al. (2023), which filtered the D-form samples after they were generated.

Model architecture and hyperparameters. We use the same SDEs as the synthetic tasks with $\sigma = 1.0$. Due to the highly peaked and complex energy distribution, we increased the discretization steps to 100 for training and 500 for evaluation. The model architecture was the same as with synthetic tasks, but the MLP hidden dimension was increased to 1024. We increase the number of epochs to $N_{\text{epochs}} = 30000$, the batch size to $K = 2000$, and the buffer size to 2000000. We use learning rates 0.0005 for the policy u_θ and 0.05 for $\log Z_\theta$. Other hyperparameters remain the same as the synthetic target tasks.

Buffer augmentation with MCMC. While our proposed importance-weighted training showed significant improvement, results were not fully satisfactory because we didn't use gradient (force) information, unlike prior works (Zhang & Chen, 2022; Midgley et al., 2023). To further enhance performance in this challenging task, we augment the buffer with samples from a gradient-informed MCMC algorithm. This approach aligns with the local search methods introduced by Sendera et al. (2024) and further developed in Kim et al. (2025b).

Specifically, we employ underdamped Langevin dynamics tailored for molecular conformation following Kim et al. (2025b), equipped with BAOAB integration (Leimkuhler & Matthews, 2013). We run multiple independent MCMC chains starting from samples in the buffer; thus, the quality of MCMC samples depends on the buffer type.

Every 500 epochs, we run 100 MCMC chains for 500 steps each. We intentionally use a small number of iterations to ensure that performance gains are not solely due to MCMC. Over 30000 epochs, this results in 3 million additional energy evaluations, which is relatively small compared to the approximately 20 million energy calls from the sampler.

C. Evaluation Metrics

In this section, we introduce the evaluation metrics used to benchmark the diffusion samplers. We employ three distribution-based metrics: the evidence lower bound (ELBO), importance-weighted ELBO (IW-ELBO), evidence upper bound (EUBO; Blessing et al., 2024). Also, following Chen et al. (2025), we incorporate the Sinkhorn distance (Cuturi, 2013) as a sample-based metric. Note that EUBO and Sinkhorn distance calculations require unbiased samples from the target distribution.

ELBO. The ELBO is defined as

$$\begin{aligned}\text{ELBO} &= \mathbb{E}_{\tau=(\mathbf{x}_0\to\cdots\to\mathbf{x}_1)\sim q_\theta}\left[\log \frac{\exp(-\mathcal{E}(\mathbf{x}_{t_N}))p_\phi(\mathbf{x}_{t_{0:N-1}}|\mathbf{x}_{t_N})}{q_\theta(\mathbf{x}_{t_{0:N}})}\right] \\ &= \mathbb{E}_{\tau\sim q_\theta}\left[\log \frac{\exp(-\mathcal{E}(\mathbf{x}_1))p_\phi(\tau \mid \mathbf{x}_1)}{q_\theta(\tau)}\right],\end{aligned}$$

where $\mathbf{x}_1$ is the final state of trajectory τ. The ELBO is a lower bound on the true log-partition function $\log Z$. We use a Monte Carlo estimate using 10,000 samples from q_θ for synthetic tasks and 100,000 for alanine dipeptide.

IW-ELBO. The K-sample IW-ELBO is an importance-weighted variant of the ELBO, defined as

$$\text{IW-ELBO}_K = \mathbb{E}_{\tau^1,\ldots,\tau^K\sim q_\theta}\left[\log \frac{1}{K}\sum_{i=1}^{K} \frac{\exp(-\mathcal{E}(\mathbf{x}_1^i))p_\phi(\tau^i \mid \mathbf{x}_1^i)}{q_\theta(\tau^i)}\right],$$

where $\mathbf{x}_1^i$ is the final state of trajectory τ^i. Notice that $K = 1$ recovers the ELBO. This is a lower bound on $\log Z$ for all K and approaches $\log Z$ as $K \to \infty$ assuming q_θ has full support (Burda et al., 2016); in fact:

$$\text{ELBO} = \text{IW-ELBO}_1 \le \text{IW-ELBO}_2 \le \cdots \le \text{IW-ELBO}_{K-1} \le \text{IW-ELBO}_K \xrightarrow{K\to\infty} \log Z.$$

We report a Monte Carlo estimate of IS-ELBO using a single batch of K trajectories, where K is the same as used to estimate the ELBO.

EUBO. The EUBO is defined as

$$\text{EUBO} = \mathbb{E}_{\mathbf{x}_1 \sim \pi, \tau \sim p_\phi(\tau|\mathbf{x}_1)} \left[\log \frac{\exp(-\mathcal{E}(\mathbf{x}_1)) p_\phi(\tau \mid \mathbf{x}_1)}{q_\theta(\tau)} \right]$$

and is an upper bound on $\log Z$. We estimate EUBO with K samples as above.

Sinkhorn distance. Sinkhorn distance is the entropic optimal transport cost, using the squared-Euclidean distance and regularization parameter 1, between two batches of K samples: one from the ground truth and one from a trained sampler. We use JAX `ott` library (Cuturi et al., 2022) to compute the Sinkhorn distance.