SINGLE-STEP CONSISTENT DIFFUSION SAMPLERS

Pascal Jutras-Dubé, Patrick Pynadath & Ruqi Zhang
Department of Computer Science
Purdue University
West Lafayette, IN 47906, USA
`{pjutrasd,ppynadat,ruqiz}@purdue.edu`

ABSTRACT

Sampling from unnormalized target distributions is a fundamental yet challenging task in machine learning and statistics. Existing sampling algorithms typically require many iterative steps to produce high-quality samples, leading to high computational costs that limit their practicality in time-sensitive or resource-constrained settings. In this work, we introduce *consistent diffusion samplers*, a new class of samplers designed to generate high-fidelity samples in a single step. We first develop a distillation algorithm to train a consistent diffusion sampler from a pretrained diffusion model without pre-collecting large datasets of samples. Our algorithm instead leverages incomplete sampling trajectories and noisy intermediate states directly from the diffusion process. We further propose a method to train a consistent diffusion sampler from scratch, fully amortizing exploration by training a single model that both performs diffusion sampling and skips intermediate steps using a self-consistency loss. Through extensive experiments on a variety of unnormalized distributions, we show that our approach yields high-fidelity samples using less than 1% of the network evaluations required by traditional diffusion samplers.

1 INTRODUCTION

Sampling from densities of the form $p_{\text{target}} = \rho/Z$ with ρ evaluable pointwise but Z intractable, is a central problem in machine learning (Neal, 1995; Hernández-Lobato & Adams, 2015) and statistics (Neal, 2001; Andrieu et al., 2003), and has applications in scientific fields like physics (Wu et al., 2019; Albergo et al., 2019; Noé et al., 2019), chemistry (Frenkel & Smit, 2002; Hollingsworth & Dror, 2018; Holdijk et al., 2024), and many other fields involving probabilistic models. Many established sampling algorithms are inherently iterative, with the accuracy of the final samples depending heavily on the number of steps. Classical Markov chain Monte Carlo (MCMC) methods asymptotically converge to the target distribution as the number of steps goes to infinity(MacKay, 2003; Robert, 1995), while more recent diffusion-based approaches (Zhang & Chen, 2022; Vargas et al., 2023; Berner et al., 2024) guarantee convergence in a finite number of steps but often necessitate hundreds of iterations to yield high-quality samples. Such iterative samplers tend to suffer from slow mixing, making them impractical for use in large models and resource-limited scenarios.

Recent work on diffusion generative models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song & Ermon, 2019; Song et al., 2021) have proposed fewer-step sampling via knowledge distillation (Salimans & Ho, 2022; Song et al., 2023). However, directly applying these distillation techniques to unnormalized distributions is challenging, as it often requires large datasets of samples that may be expensive to collect. This motivates the following question of wether we can significantly reduce the steps required by samplers, enabling few-step or even single-step sampling.

In this paper, we propose *consistent diffusion samplers* to produce high-quality samples in a single step. We first show that diffusion-based samplers can be *consistently distilled* into single-step diffusion samplers. Instead of storing a large dataset of fully diffused samples, our approach exploits incomplete trajectories and noisy samples encountered during the diffusion process. We further introduce a *self-consistent* diffusion sampler that does not require a pretrained diffusion sampler. Instead, it fully amortizes exploration by jointly learning both diffusion sampling and large cut off steps that match the outcome of paths of small steps. This enables single-step sampling yet retains

the option to refine samples through multiple iterations if desired, subsuming existing diffusion-based approaches.

Our contributions can be summarized as follows:

- We show that diffusion-based samplers for unnormalized distributions can be effectively distilled into single-step consistent samplers without pre-collecting large datasets of samples.

- We introduce a self-consistent diffusion sampler that learns to perform single-step sampling by jointly training diffusion-based transitions and large shortcut steps via a self-consistency criterion. This method only trains one neural network and does not require pretrained samplers or high-quality data.

- Through extensive evaluations on synthetic and real unnormalized distributions, we demonstrate that our method delivers competitive sample quality while drastically reducing sampling steps.

2 RELATED WORK

Markov chain Monte Carlo (MCMC). Markov chain Monte Carlo methods are a classical approach for sampling from unnormalized target densities. The key idea is to construct a Markov chain whose stationary distribution matches the target distribution (Brooks et al., 2012). Prominent examples include the Metropolis-Hastings algorithm (Metropolis et al., 1953; Hastings, 1970), Gibbs sampling (Geman & Geman, 1984), and Langevin dynamics (Rossky et al., 1978; Parisi, 1981). By exploiting geometric structure in the target distribution, Hamiltonian Monte Carlo (Duane et al., 1987; MacKay, 2003; Brooks et al., 2012; Chen et al., 2014) often leads to more efficient exploration. To address scalability challenges in high-dimensional or large-dataset scenarios, stochastic gradient MCMC variants (Welling & Teh, 2011; Chen et al., 2014; Zhang et al., 2020a;b) have been introduced. Although these MCMC methods reduce per-step computational costs or improve mixing, they remain inherently iterative, requiring many transitions to yield high-quality samples.

Learning-Based Samplers. Amortized inference shifts the computational overhead from test-time sampling to a training phase, allowing for faster inference (Gershman & Goodman, 2014). Approaches such as amortized MCMC (Li et al., 2017) train a neural network to mimic the distribution of samples obtained after T transitions of a traditional MCMC process. Similarly, GFlowNets (Bengio et al., 2021; 2023) learn to sequentially construct complex discrete objects, effectively learning a sampling strategy. While GFlowNets amortize the computational challenges of lengthy stochastic searches and mode-mixing during training, their sampling process remains sequential, as objects are constructed step-by-step through a series of constructive steps.

An alternative viewpoint casts the sampling problem as an optimal control task (Zhang & Chen, 2022; Berner et al., 2024; Richter & Berner, 2024), where one trains a controlled stochastic differential equation to transport an initial distribution to the target via a Schrödinger bridge (Schrödinger, 1931; 1932). This perspective motivates recent efforts to use diffusion-based samplers (Geffner & Domke, 2023; Vargas et al., 2023; Zhang et al., 2024; Phillips et al., 2024; Chen et al., 2025). While such diffusion and flow-based frameworks have advanced the state of the art, they require numerical solvers operating on fine time discretizations.

Consistent Generative Models. Recent work in generative modeling has explored the concept of consistency: ensuring that large transitions between observed distributions are consistent with sequences of incremental transformations. Consistency models (Song et al., 2023; Song & Dhariwal, 2023; Lu & Song, 2025) learn a direct mapping from any point in time to the terminal state. Progressive distillation (Salimans & Ho, 2022; Meng et al., 2023) incrementally distills a trained diffusion model into a more efficient version that takes half as many until a single-step model is achieved. Similarly, shortcut models (Liu et al., 2023; Frans et al., 2025) leverage progressive self-distillation during training to achieve accelerated inference without relying on a pre-trained teacher model.

These methods focus on generative modeling tasks and assume access to a dataset drawn from the target distribution. Our work introduces the notion of consistency into the setting of sampling from unnormalized densities. We assume access only to an unnormalized pointwise oracle ρ for the target density, without requiring any pre-collected samples.

3 PRELIMINARIES: DIFFUSION-BASED SAMPLERS

Diffusion-based samplers are controlled stochastic differential equations (SDEs) that transport samples from a simple prior distribution p_{prior} to the target distribution p_{target}. Consider a forward-time SDE over $t \in [0, T]$ with initial condition $\mathbf{x}_0 \sim p_{\text{prior}}$:

$$\mathrm{d}\mathbf{x}_t = \big(\mu(t)\mathbf{x}_t + g(t)u_\theta(\mathbf{x}_t, t)\big)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}_t, \tag{1}$$

where $\mathbf{w}$ is a standard Brownian motion, $\mu(\cdot)\mathbf{x}_t$ is the drift term, g is the diffusion coefficient, and u_θ is a learned control term parameterized by a neural network.

Further consider the time-reversal process $\mathbf{y}$ of a diffusion that gradually adds noise to samples from the target distribution:

$$\mathrm{d}\mathbf{y}_t = \big(\mu(t)\mathbf{y}_t + g^2(t)\nabla \log p_{\mathbf{y}_t}(\mathbf{y}_t)\big)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}_t. \tag{2}$$

If we choose $\mathbf{y}_0 \sim p_{\text{prior}}$ and μ and g such that $\mathbf{y}_T \sim p_{\text{target}}$, then setting $u_\theta(\mathbf{x}_t, t) = g(t)\nabla \log p_{\mathbf{y}_t}(\mathbf{x}_t)$ in Eq. 1 would yield $p_{\mathbf{x}_t} = p_{\mathbf{y}_t}$ and thus $\mathbf{x}_T \sim p_{\text{target}}$ (Anderson, 1982). In practice, however, the score function $\nabla \log p_{\mathbf{y}_t}$ is unknown and must be approximated by training u_θ.

Let $\mathbb{P}_{\mathbf{x}}$ denote the path space measure induced by the SDE in Eq. 1, and $\mathbb{P}_{\mathbf{y}}$ the path space measure for the time-reversed process in Eq. 2. Further, let $\mathcal{U} \subset C\big(\mathbb{R}^d \times [0, T], \mathbb{R}^d\big)$ be a space of admissible controls. From an optimal control and path space perspective (Berner et al., 2024; Richter & Berner, 2024), the diffusion sampling problem can be framed as finding an optimal control u^* that minimizes a divergence between these two path measures:

$$u^* \in \arg\min_{\mathcal{U}} D(\mathbb{P}_{\mathbf{x}} \,\|\, \mathbb{P}_{\mathbf{y}}), \tag{3}$$

where $D(\cdot \,\|\, \cdot)$ is an appropriate divergence. To evaluate $D(\mathbb{P}_{\mathbf{x}} \,\|\, \mathbb{P}_{\mathbf{y}})$, one requires the Radon–Nikodym derivative, which measures how much more likely a given trajectory $\mathbf{v}$ is under $\mathbb{P}_{\mathbf{x}}$ than under $\mathbb{P}_{\mathbf{y}}$:

$$\frac{\mathrm{d}\mathbb{P}_{\mathbf{x}}}{\mathrm{d}\mathbb{P}_{\mathbf{y}}}(\mathbf{v}) = Z\exp\big(R(\mathbf{v}) + S(\mathbf{v}) + B(\mathbf{v})\big) \tag{4}$$

where

$$R(\mathbf{x}) = \int_0^T \big(\tfrac{1}{2}\|u_\theta(\mathbf{x}_t, t)\|^2 - \mathrm{div}(\mu(t)\mathbf{x}_t)\big)\,\mathrm{d}t, \quad S(\mathbf{x}) = \int_0^T u_\theta(\mathbf{x}_t, t) \cdot \mathrm{d}\mathbf{w}_t,$$

$$\text{and } B(\mathbf{x}) = \log\frac{p_{\text{prior}}(\mathbf{x}_0)}{\rho(\mathbf{x}_T)}.$$

Two widely used divergences in diffusion-based sampling are $D_{\text{KL}}(\mathbb{P}_{\mathbf{x}} \,\|\, \mathbb{P}_{\mathbf{y}}) = \mathbb{E}\big[R(\mathbf{x}) + B(\mathbf{x})\big] + \log Z$ and $D_{\text{LV}}(\mathbb{P}_{\mathbf{x}} \,\|\, \mathbb{P}_{\mathbf{y}}) = \mathbb{V}\big[R(\mathbf{x}) + S(\mathbf{x}) + B(\mathbf{x})\big]$. Here, D_{KL} is the Kullback–Leibler divergence (Zhang & Chen, 2022; Vargas et al., 2023; Berner et al., 2024), and D_{LV} is the log-variance divergence (Nüsken & Richter, 2021; Richter & Berner, 2024).

Once trained, the control u_θ allows for generating samples from p_{target} by simulating the forward SDE Eq. 1. In practice, numerical discretization $0 = t_1 < t_2 < \ldots < t_N = T$ is required, and finer time steps yield more accurate sampling but at higher computational cost. Thus, a key challenge lies in balancing step size against the desired accuracy and efficiency.

4 CONSISTENCY DISTILLED DIFFUSION SAMPLERS

In this section, we show how to adapt consistency distillation to the problem of sampling from unnormalized densities. We name our method the *consistency distilled diffusion sampler* (CDDS). The next section will address how to remove the requirement of having a pre-trained diffusion sampler.

Our goal is to learn a consistency function $f : (\mathbf{x}_t, t) \mapsto \mathbf{x}_T$, which maps any intermediate state $\mathbf{x}_t$ directly to a sample $\mathbf{x}_T$ from the target distribution. Although we lack a dataset of samples from p_{target}, if we possess a pre-trained diffusion sampler, we can approximate such a dataset

---

Algorithm 1 Data-Free Consistency Distillation

Input: model parameters $\boldsymbol{\theta}$, control u, learning rate η, distance d, weight λ
$\boldsymbol{\theta}' \leftarrow \boldsymbol{\theta}$
repeat
    Sample $\mathbf{x}_0 \sim p_{\text{prior}}$ and $n \sim \mathcal{U}\{1, N-1\}$
    Integrate Eq. (5) to obtain $\hat{\mathbf{x}}_{t_n}$ and $\hat{\mathbf{x}}_{t_{n+1}}$
    $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}'; u) \leftarrow \lambda(t_n) d\big(f_{\boldsymbol{\theta}'}(\hat{\mathbf{x}}_{t_{n+1}}, t_{n+1}), f_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}, t_n)\big)$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla \boldsymbol{\theta} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}'; u)$
    $\boldsymbol{\theta}' \leftarrow \text{stopgrad}(\boldsymbol{\theta})$
until convergence

---

by simulating the generative SDE Eq. 1, producing samples $\{\hat{\mathbf{x}}_T^i\}_{i=1}^M$. We can then apply either consistency distillation or consistency training (as in Algorithms 2 and 3 of Song et al., 2023) to learn f. This approach is expensive as it necessitates pre-collecting and storing a large dataset.

Consider a pre-trained diffusion process whose trajectories $\mathbf{x}_{t_1}, \mathbf{x}_{t_2}, \ldots, \mathbf{x}_T$ would normally be used to create a dataset for distillation. Instead, we directly leverage intermediate states $\mathbf{x}_t$ during each training iteration. This reduces storage demands and limits the accumulation of numerical errors that could arise from fully integrating the numerical solver. If the error per step of an order-p solver is bounded by $O((t_{n+1} - t_n)^{p+1})$, using multiple, shorter intervals can help keep the overall global error smaller.

To learn a deterministic mapping $(\mathbf{x}_t, t) \mapsto \mathbf{x}_T$, we simulate the associated probability flow (PF) ODE (Song et al., 2021):

$$d\mathbf{x}_t = \Big(\mu(\mathbf{x}_t, t) + \tfrac{1}{2}\sigma(t), u(\mathbf{x}_t, t)\Big) dt, \tag{5}$$

We minimize the discrepancy between the outputs of the consistency function at consecutive intermediate states:

$$\mathcal{L}_{\text{CD}}(\boldsymbol{\theta}, \boldsymbol{\theta}'; u) := \mathbb{E}\Big[\lambda(t_n) d\big(f_{\boldsymbol{\theta}'}(\hat{\mathbf{x}}_{t_{n+1}}, t_{n+1}), f_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_n}, t_n)\big)\Big], \tag{6}$$

where $d(\cdot, \cdot)$ is a distance metric, $\lambda(\cdot)$ is a positive weighting function, and $\boldsymbol{\theta}' = \text{stopgrad}(\boldsymbol{\theta})$ indicates that the gradients are not passed through the target term. Notably, different to training consistency generative models, here, both $\hat{\mathbf{x}}_{t_{n+1}}$ and $\hat{\mathbf{x}}_{t_n}$ are approximate states obtained by partially integrating the PF ODE. Training a consistent diffusion sampler via distillation requires a similar computational cost as training the original diffusion sampler, since both processes involve simulating trajectories; however, it enables faster inference at test time. The training procedure is summarized in Algorithm 1.

If the loss in Eq. 6 is driven to zero, the learned consistency function can approximate the true mapping arbitrarily well, provided the step size of the ODE solver is sufficiently small. We formally state this in Theorem 4.1.

Theorem 4.1. *Let $\boldsymbol{f}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ be a consistency function parameterized by $\boldsymbol{\theta}$, and let $\boldsymbol{f}(\mathbf{x}_t, t; u)$ denote the consistency function of the PF ODE defined by the control u. Assume that $\boldsymbol{f}_{\boldsymbol{\theta}}$ satisfies a Lipschitz condition with constant $L > 0$, such that for all $t \in [0, T]$ and for all $\mathbf{x}_t, \mathbf{y}_t$,*

$$\|\boldsymbol{f}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \boldsymbol{f}_{\boldsymbol{\theta}}(\mathbf{y}_t, t)\|_2 \leq L \|\mathbf{x}_t - \mathbf{y}_t\|_2.$$

Additionally, assume that for each step $n \in \{1, 2, \ldots, N-1\}$, the ODE solver called at t_n has a local error bounded by $O((t_{n+1} - t_n)^{p+1})$ for some $p \geq 1$.

If, additionally, $\mathcal{L}_{CD}(\boldsymbol{\theta}, \boldsymbol{\theta}; u) = 0$, then:

$$\sup_{n, \mathbf{x}_{t_n}} \|\boldsymbol{f}_{\boldsymbol{\theta}}(\mathbf{x}_{t_n}, t_n) - \boldsymbol{f}(\mathbf{x}_{t_n}, t_n; u)\|_2 = O((\Delta t)^p),$$

where $\Delta t := \max_{n \in \{1,2,\ldots,N-1\}} |t_{n+1} - t_n|$. A complete proof is provided in Appendix B.

While our distillation approach builds upon the core principles of consistency models, it differs in setting and requirements. Consistency generative models assume direct access to real samples from the target distribution. In contrast, our consistency distilled diffusion samplers address the problem of sampling from unnormalized target densities, where no dataset of target samples is available. Our method extends consistency distillation to sampling from unnormalized distributions, making it applicable beyond generative modeling tasks.

5 SELF-CONSISTENT DIFFUSION SAMPLERS

In this section, we introduce *self-consistent diffusion sampler* (SCDS) that achieves single-step sampling without requiring a pre-trained diffusion sampler. Our motivation stems from merging two complementary perspectives.

First, diffusion-based samplers learn a time-dependent control function that steers an SDE from a simple prior distribution to the target distribution. Typically, the control is trained on a fixed schedule (e.g., N small increments of length T/N along a discretized time axis), requiring multiple steps. Second, consistency models learn a direct mapping from any intermediate state on an ODE to the terminal state. In other words, at time t the model is implicitly taught to jump a large step of length $T - t$.

Our idea is to unify these approaches in a single model. Specifically, we condition a control function $u_\theta(\mathbf{x}_t, t, d)$ on both the current time t and the desired step size d. By adjusting d, the model can adapt between short incremental steps (as in standard diffusion samplers) and large jumps (as in consistency models). This design amortizes the learning of both small and large transitions into one network and recovers consistency models' single-step sampling by setting $d = T - t$ and diffusion sampling by setting $d = T/N$. In doing so, we avoid training two separate models.

Enforcing Self-Consistency. To ensure that the step-size-conditioned control function $u_\theta(\mathbf{x}_t, t, d)$ remains accurate across varying step sizes, we introduce a self-consistency loss. The key idea is that taking a large step should yield the same result as taking multiple smaller steps. To do so, we impose a consistency condition on the Euler discretization of the PF ODE Eq. 5. Specifically, a single large step of size $2d$,

$$\mathbf{x}_{t+2d} = \mathbf{x}_t + \left(\mu(t)\mathbf{x}_t + \tfrac{1}{2}g(t)u_\theta(\mathbf{x}_t, t, 2d)\right) 2d, \tag{7}$$

must equal two smaller steps of size d. The intermediate state is computed as

$$\mathbf{x}'_{t+d} = \mathbf{x}_t + \left(\mu(t)\mathbf{x}_t + \tfrac{1}{2}g(t)u_{\theta'}(\mathbf{x}_t, t, d)\right) d$$

and the final state after two steps is

$$\mathbf{x}'_{t+2d} = \mathbf{x}'_{t+d} + \left(\mu(t+d)\mathbf{x}_{t+d} + \tfrac{1}{2}g(t+d)u_{\theta'}(\mathbf{x}_{t+d}, t+d, d)\right) d, \tag{8}$$

where $\theta' = \text{stopgrad}(\theta)$. The self-consistency objective is a simple least square minimization problem:

$$\mathcal{L}_{\text{SC}} = \mathbb{E}\left[\left\|\mathbf{x}'_{t+2d} - \mathbf{x}_{t+2d}\right\|^2\right] \tag{9}$$

where the expectation is taken over time indices and step sizes drawn from the simulated trajectories.

This loss encourages the model to correct for numerical errors when taking large steps, allowing it to "skip" multiple smaller steps while remaining consistent with the dynamics of the PF ODE. To initiate this recursive training, we must define and learn the behavior at the base case $d = T/N$.

Learning the Base Case $d = T/N$. In standard generative modeling scenarios (where a dataset is available), the base case $d = T/N$ can be learned directly from data using deterministic trajectories (Lipman et al., 2023; Frans et al., 2025). These trajectories provide explicit guidance toward high-density regions of the target distribution.

However, when working with an unnormalized density, the key challenge is discovering high-probability regions (modes). In such cases, exploration is necessary to locate and model these regions effectively (Chen et al., 2025). Diffusion-based samplers facilitate exploration through their stochastic dynamics: Brownian motion helps probe different parts of the space, allowing the model to learn and adapt itself to the target distribution.

Thus, diffusion-based sampling is particularly well-suited for learning the base case. The running term R in the Radon-Nikodym derivative 4 requires simulating the stochastic process in Eq. 1, allowing it to learn the structure of high-density regions. In this work, we adopt the log-variance divergence as our base sampling objective:

$$\mathcal{L}_{\text{S}} = D_{\text{LV}}(\mathbb{P}_{\mathbf{x}} \| \mathbb{P}_{\mathbf{y}}). \tag{10}$$

By optimizing $u_\theta(\mathbf{x}_t, t, d = T/N)$ under this loss, we ensure that the model can generate meaningful transitions from the prior to these regions of interest, forming a strong foundation for self-consistent learning at larger step sizes.
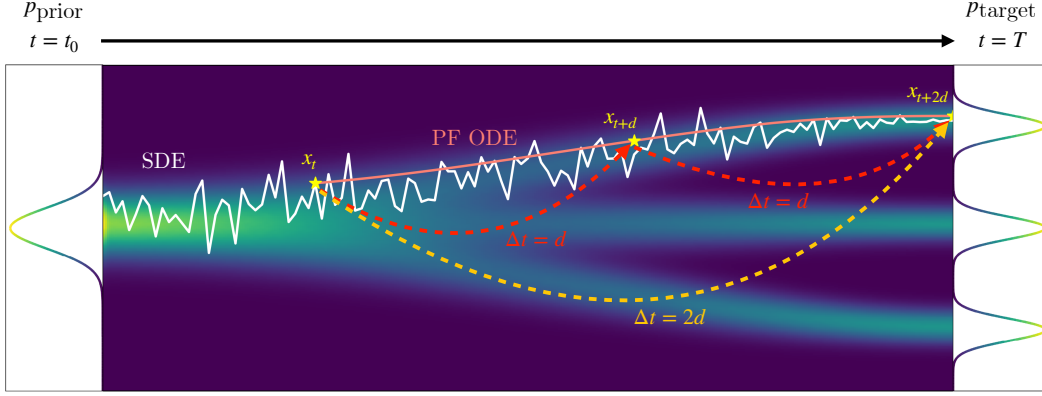
Figure 1: Graphical illustration of the training procedure for SCDS over the path space. First, the SDE trajectory (white) is simulated to compute the sampling loss $\mathcal{L}_S$. Next, a timestep t and a step size d are randomly sampled. From $\mathbf{x}_t$ on the simulated SDE trajectory, we execute two consecutive steps of size d (red) along the PF-ODE trajectory (pink), obtaining the target $\mathbf{x}'_{t+2d}$. Finally, the shortcut step of size d (orange) predicts $\mathbf{x}_{t+2d}$ directly from $\mathbf{x}_t$, and the self-consistency loss $\mathcal{L}_{SC}$ minimizes the squared difference between $\mathbf{x}_{t+2d}$ and the two-step target $\mathbf{x}'_{t+2d}$, ensuring multi-scale consistency.

Algorithm 2 SCDS Training

Input Model parameters $\boldsymbol{\theta}$, loss weightings $\lambda_S(\cdot)$ and $\lambda_{SC}(\cdot)$
$\boldsymbol{\theta}' \leftarrow \boldsymbol{\theta}$
repeat
    Sample $\mathbf{x}_0 \sim p_{\text{prior}}$ and $(d, t) \sim p_{d,t}$.
    Compute $\mathbf{x} \leftarrow (\mathbf{x}_i)_{i=0}^T$ by simulating Eq. 1
    Compute $\mathbf{x}'_{t+2d}$ from Eq. 7
    Compute $\mathbf{x}_{t+2d}$ from Eq. 8
    Compute $\mathcal{L}_S$ using Eq. 10
    Compute $\mathcal{L}_{SC}$ using Eq. 9.
    $\boldsymbol{\theta} \leftarrow \nabla_{\boldsymbol{\theta}} (\lambda_S(t)\mathcal{L}_S + \lambda_{SC}(t)\mathcal{L}_{SC})$
    $\boldsymbol{\theta}' \leftarrow \text{stopgrad } \boldsymbol{\theta}$
until convergence

End to End Training Algorithm. Our training procedure jointly optimizes two objectives: (1) the sampling loss Eq. 10 for the base case $d = T/N$, which ensures exploration and score approximation by simulating the SDE Eq. 1, and (2) the self-consistency loss Eq. 9 enforced on the PF-ODE Eq. 5 for larger d, which enforces consistency across multiple time scales.

To enable the recursive halving of steps, we discretize the time interval $[0, T]$ into $N + 1$ points, where N is chosen as a power of two. The sampling loss is computed by simulating the forward SDE along this time grid. For self-consistency training, we sample step sizes d and times t such that d are powers of two (multiplied by T/N) dividing the remaining time $T - t$. This ensures that from any time t, we can take exactly k steps of size d to reach the terminal state for some integer k. This way, training focuses on time sequences that are applicable during inference.

To compute the self-consistency loss, we extract $\mathbf{x}_t$ from the simulated forward SDE. Using $\mathbf{x}_t$ and the sampled step size d, we compute the shortcut step $\mathbf{x}_{t+2d}$ using Eq. 7 and the two-step target trajectory $\mathbf{x}'_{t+2d}$ using Eq. 8 on the PF ODE. We then optimize their squared difference via Eq. 9, ensuring that larger steps remain consistent with fine-grained trajectories. The training procedure is summarized in Algorithm 2 and illustrated in Figure 1. Compared to previous diffusion-based samplers, our method only incurs 3 additional network function evaluations per training iteration.

Few-step Sampling. With a well-trained control u_θ, sampling can be performed in a single step by drawing from the prior and applying a single Euler update with step size $d = T$. This accelerates generation compared to traditional diffusion-based samplers. Alternatively, our method provides a flexible tradeoff between computational efficiency and sample quality, allowing for multi-step

---

Algorithm 3 Multi-Step Sampling with SCDS

    Input: Trained model u_θ, number of sampling steps N
    Sample $\mathbf{x}_0 \sim p_{\text{prior}}$
    Initialize $d \leftarrow T/N$ and $t \leftarrow 0$
    for $k = 1, \ldots, N$ **do**
        Compute $\mathbf{x}_{t+d} = \mathbf{x}_t + \left(\mu(t)\mathbf{x}_t + \frac{1}{2}g(t)u_\theta(\mathbf{x}_t, t, d) \right) d$
        Update $t \leftarrow t + d$
    end for
    Return $\mathbf{x}_T$

Table 1: Comparison of different methods in terms of Sinkhorn distances (lower is better). We present results on tasks where ground-truth samples are available for evaluation. "NFE" refers to the number of function evaluations.

Sinkhorn (↓)		Target Distribution				
Sampler	*NFE*	*GMM (2d)*	*Image (2d)*	*Funnel (10d)*	*MW54 (5d)*	*MW52 (50d)*
SCDS *(Ours)*	128	0.0204	0.0169	5.2569	0.1191	7.4557
	2	0.0279	0.0294	5.3488	0.1955	11.5200
	1	0.0330	0.0322	5.3729	0.2102	7.4925
CDDS *(Ours)*	2	0.0241	0.0309	7.1329	0.1570	6.5010
	1	0.0224	0.0309	7.2159	0.1569	6.5285
PIS	128	0.6656	0.9168	5.9956	0.1223	7.2955
DDS	128	0.0709	1.5818	6.0467	0.1190	7.2842
DIS	128	0.0203	0.0170	5.1578	0.1197	7.3668
DIS	1	0.0551	0.2781	10.4033	6.4679	31.7883

refinement when needed, thus recovering standard diffusion-based sampling. This iterative procedure is detailed in Algorithm 3.

Approximating Z. A benefit of SCDS is the ability to estimate the intractable normalizing constant Z. By leveraging the relationship established in the RN derivative (Eq. 4), we can approximate $\log Z$. Specifically, when the optimal control $u^* = g(t)\nabla \log p_{\mathbf{y}_t}(\mathbf{x}_t)$ is attained, the KL divergence $D_{\text{KL}}(\mathbb{P}_\mathbf{x} \| \mathbb{P}_\mathbf{y})$ reaches zero. This implies $-\log Z = \min_{u \in \mathcal{U}} \mathbb{E}[R(\mathbf{x}) + B(\mathbf{x})]$. Unlike CDDS and consistency models, which focus on solely sample generation, SCDS leverages the control-based formulation to handle both sampling and the normalizing constant, making it applicable to a broader range of probabilistic tasks.

Learning Shortcuts Without Data. SCDS shares conceptual similarities with progressive distillation (Salimans & Ho, 2022) and shortcut models (Frans et al., 2025), both of which enforce that a large time step transition should be consistent with two half-sized transitions. However, these methods rely on access to a dataset or to a pre-trained teacher model. In contrast, SCDS learns both the diffusion process and shortcut connections directly from an unnormalized density.

6 EXPERIMENTS

Experimental Setup. We evaluate our CDDS and SCDS on multiple sampling benchmarks: a 9-mode Gaussian mixture model in 2d (GMM), a 2d image of a labrador (Image), a 10d Funnel distribution, and two 32-mode many-well tasks (MW54 in 5d and MW52 in 50d). We also consider a high-dimensional log Gaussian Cox Process (LGCP) problem in 1600d. We compare to path integral sampler (PIS) (Zhang & Chen, 2022), denoising diffusion sampler (DDS) (Vargas et al., 2023), and time-reversed diffusion sampler (DIS) (Berner et al., 2024). We also show a single-step version of DIS as a naive baseline to gauge how single-step sampling might upper-bound the Sinkhorn distance if we remove shortcuts. In our experiments, CDDS is a distilled version of DIS. We use Fourier features network to condition SCDS on the stepsize d (Tancik et al., 2020). When ground-truth samples are available, we measure performance via the Sinkhorn distance (Cuturi, 2013) between generated samples and samples from the target distribution. For the LGCP task, we report the relative
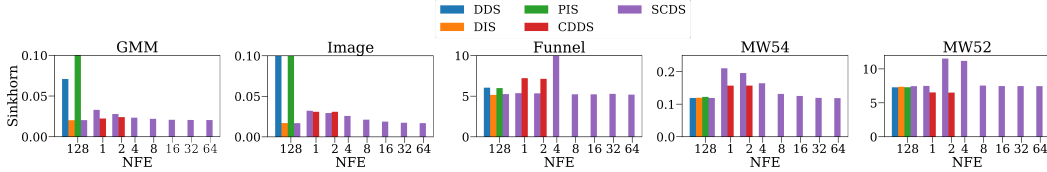
Figure 2: Comparison of Sinkhorn distance for a range of NFEs between the proposed consistency samplers (CDDS, SCDS) and diffusion-based samplers (PIS, DDS, DIS). For most targets, CDDS and SCDS show competitive Sinkhorn values with baselines with much lower NFEs.

error of the estimated normalizing constant $\log Z$. Additionally, we quantify the number of function evaluations (NFE) (Karras et al., 2022). For more details on the training of the various samplers, along with evaluation details and target distribution settings, see Appendix C.

Sinkhorn Results and Analysis. Table 1 shows that both CDDS and SCDS maintain competitive sinkhorn distances in single- and two-steps generations compared to existing diffusion-based samplers with 128 steps. With only one step, SCDS ans CDDS consistently outperform single-step DIS by a clear margin on every task, highlighting the benefits of enforcing consistency. As with other consistency-based methods Song et al. (2023) we find CDDS's multi-step performance typically saturates after 2–3 steps, indicating minimal gains from iterative refinements. In contrast, SCDS's accuracy steadily improves with increasing step counts in most tasks (see Figure 2), except for minor dips at 4 steps in Funnel and at 2/4 steps in MW52. The general upward trend demonstrates that SCDS effectively recovers standard multi-step diffusion behaviors.

Log Gaussian Cox Process. Table 2 compares $\log Z$ estimation errors for each method on the 1600d LGCP task. Multi-step PIS and DIS achieve smaller errors then SCDS, but SCDS remains viable even at reduced NFEs. Notably, as expected, single-step DIS fails catastrophically, whereas single-step SCDS remains stable. Since SCDS learns a time-dependent control function, it retains a connection to the Radon-Nikodym derivative in Eq. 4, allowing for partition function estimation. In contrast, CDDS (and consistency models in general) lack an explicit control representation, meaning they cannot directly estimate Z. This is a key advantage of SCDS in applications where unnormalized densities must be integrated, such as Bayesian inference.

Discussion. A key advantage of SCDS lies in its ability to learn both the diffusion sampling process and the self-consistency shortcuts *simultaneously*. In contrast to consistency models, which require a pre-trained sampler or high-fidelity trajectories for distillation, SCDS forgoes such prerequisite. SCDS is often competitive with well-established diffusion samplers and consistency-distilled approach CDDS that benefit from a carefully tuned, pre-trained teacher. Moreover, SCDS adapts seamlessly from single-step to many-step sampling without retraining.

Table 2: Relative error of Log Z estimates for various samplers on LGCP target distribution.

LGCP (1600d)		
Sampler	*NFE*	*Log Z (LB) Error* (↓)
SCDS (Ours)	128	0.9968
	64	1.0506
	32	1.5976
	16	2.2378
	8	2.7931
	4	3.9660
	2	6.2420
	1	9.9877
PIS	128	0.2910
DDS	128	2.8545
DIS	128	0.3736
	1	3094.7296

7 CONCLUSION

We introduced two novel approaches for efficient sampling from unnormalized target distributions: *consistency-distilled diffusion samplers* (CDDS) and the *self-consistent diffusion sampler* (SCDS). CDDS uses consistency distillation without generating a large dataset of samples. SCDS requires no pre-trained samplers and simultaneously learns to sample high-density regions and to take large steps across the path space. Our empirical results across a range of benchmarks demonstrate that both methods achieve competitive accuracy with as few as one or two steps. These findings highlight the potential of consistency-based methods for sampling from unnormalized densities.

REFERENCES

Michael S Albergo, Gurtej Kanwar, and Phiala E Shanahan. Flow-based generative models for markov chain monte carlo in lattice field theory. *Physical Review D*, 100(3):034515, 2019.

Brian. D. O. Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12:313–326, 1982.

Christophe Andrieu, Nando de Freitas, Arnaud Doucet, and Michael I. Jordan. An introduction to mcmc for machine learning. *Machine Learning*, 50(1-2):5–43, 2003.

Michael Arbel, Alex Matthews, and Arnaud Doucet. Annealed flow transport monte carlo. In *International Conference on Machine Learning*, pp. 318–330. PMLR, 2021.

Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow network based generative models for non-iterative diverse candidate generation. In *Advances in Neural Information Processing Systems*, 2021.

Yoshua Bengio, Salem Lahlou, Tristan Deleu, Edward J. Hu, Mo Tiwari, and Emmanuel Bengio. Gflownet foundations. *Journal of Machine Learning Research*, 24(210):1–55, 2023.

Julius Berner, Lorenz Richter, and Karen Ullrich. An optimal control perspective on diffusion-based generative modeling. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856.

Steve P. Brooks, Andrew Gelman, Galin L. Jones, and Xiao-Li Meng. Handbook of markov chain monte carlo: Hardcover. *CHANCE*, 25:53–55, 2012.

Junhua Chen, Lorenz Richter, Julius Berner, Denis Blessing, Gerhard Neumann, and Anima Anandkumar. Sequential controlled langevin diffusions. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025.

Tianqi Chen, Emily Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *International Conference on Machine Learning*, pp. 1683–1691. PMLR, 2014.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems*, pp. 2292–2300, 2013.

Simon Duane, A.D. Kennedy, Brian J. Pendleton, and Duncan Roweth. Hybrid monte carlo. *Physics Letters B*, 195(2):216–222, 1987. ISSN 0370-2693.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. In *Proceedings of the International Conference on Learning Representations*. ICLR, 2025.

Daan Frenkel and Berend Smit. *Understanding Molecular Simulation: From Algorithms to Applications*. Academic Press, Amsterdam, The Netherlands, 2002. ISBN 978-0-12-267351-1.

Tomas Geffner and Justin Domke. Langevin diffusion variational inference. In *International Conference on Artificial Intelligence and Statistics*, pp. 576–593. PMLR, 2023.

Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6 (6):721–741, 1984.

Samuel J. Gershman and Noah D. Goodman. Amortized inference in probabilistic reasoning. In *Proceedings of the Annual Meeting of the Cognitive Science Society*. Stanford University, 2014.

W. Keith Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International Conference on Machine Learning*, pp. 1861–1869. PMLR, 2015.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851, 2020.

Lars Holdijk, Yuanqi Du, Ferry Hooft, Priyank Jaini, Berend Ensing, and Max Welling. Stochastic optimal control for collective variable free sampling of molecular transition paths. *Advances in Neural Information Processing Systems*, 36, 2024.

Scott A. Hollingsworth and Ron O. Dror. Molecular dynamics simulation for all. *Neuron*, 99(6): 1129–1143, 2018. ISSN 0896-6273.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*, 2022.

Yingzhen Li, Richard E Turner, and Qiang Liu. Approximate inference with amortised mcmc. *arXiv preprint arXiv:1702.08343*, 2017.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*, 2023.

Cheng Lu and Yang Song. Simplifying, stabilizing and scaling continuous-time consistency models. In *International Conference on Learning Representations*, 2025.

David JC MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.

Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

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.

Radford M. Neal. Bayesian learning for neural networks. 1995.

Radford M Neal. Annealed importance sampling. *Statistics and Computing*, 11:125–139, 2001.

Radford M Neal. Slice sampling. *The annals of statistics*, 31(3):705–767, 2003.

Frank Noé, Jonas Köhler, and Hao Wu. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science*, 365, 2019. URL `https://api.semanticscholar.org/CorpusID:54458652`.

Niklas Nüsken and Lukas Richter. 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.

G. Parisi. Correlation functions and computer simulations. *Nuclear Physics B*, 180(3):378–384, 1981.

Angus Phillips, Hai-Dang Dau, Michael John Hutchinson, Valentin De Bortoli, George Deligiannidis, and Arnaud Doucet. Particle denoising diffusion sampler, 2024.

Lorenz Richter and Julius Berner. Improved sampling via learned diffusions. In *International Conference on Learning Representations*, 2024.

Christian P. Robert. Convergence control methods for markov chain monte carlo algorithms. *Statistical Science*, 10(3):231–253, 1995.

P. J. Rossky, J. D. Doll, and H. L. Friedman. Brownian Dynamics as Smart Monte Carlo Simulation. *The Journal of Chemical Physics*, 69(10):4628–4633, 11 1978.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022.

Erwin Schrödinger. Über die umkehrung der naturgesetze. *Sitzungsberichte der Preussischen Akademie der Wissenschaften Berlin, Physikalisch-Mathematische Klasse*, pp. 144–153, 1931.

Erwin Schrödinger. Sur la théorie relativiste de l'Électron et l'interprétation de la mécanique quantique. *Annales de l'Institut Henri Poincaré*, 2:269–310, 1932.

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

Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. *arXiv preprint arXiv:2310.14189*, 2023.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

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

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023.

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in neural information processing systems*, 33:7537–7547, 2020.

Francisco Vargas, Will Sussman Grathwohl, and Arnaud Doucet. Denoising diffusion samplers. In *International Conference on Learning Representations*, 2023.

Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient langevin dynamics. In *International Conference on Machine Learning*, pp. 681–688, Madison, WI, USA, 2011. Omnipress. ISBN 9781450306195.

Dian Wu, Lei Wang, and Pan Zhang. Solving statistical mechanics using variational autoregressive networks. *Physical Review Letters*, 122(8):080602, 2019.

Hao Wu, Jonas Köhler, and Frank Noé. Stochastic normalizing flows. *Advances in Neural Information Processing Systems*, 33:5933–5944, 2020.

Dinghuai Zhang, Ricky Tian Qi Chen, Cheng-Hao Liu, Aaron Courville, and Yoshua Bengio. Diffusion generative flow samplers: Improving learning signals through partial trajectory optimization. In *International Conference on Learning Representations*, 2024.

Qinsheng Zhang and Yongxin Chen. Path integral sampler: A stochastic control approach for sampling. In *International Conference on Learning Representations*, 2022.

Ruqi Zhang, A Feder Cooper, and Christopher De Sa. Amagold: Amortized metropolis adjustment for efficient stochastic gradient mcmc. In *International Conference on Artificial Intelligence and Statistics*, pp. 2142–2152. PMLR, 2020a.

Ruqi Zhang, Chunyuan Li, Jianyi Zhang, Changyou Chen, and Andrew Gordon Wilson. Cyclical stochastic gradient mcmc for bayesian deep learning. *International Conference on Learning Representations*, 2020b.

A APPENDIX

B CONSISTENCY DISTILLATION PROOF

Theorem 4.1. *Let $f_\theta(\mathbf{x}_t, t)$ be a consistency function parameterized by θ, and let $f(\mathbf{x}_t, t; u)$ denote the consistency function of the PF ODE defined by the control u. Assume that f_θ satisfies a Lipschitz condition with constant $L > 0$, such that for all $t \in [0, T]$ and for all $\mathbf{x}_t, \mathbf{y}_t$,*

$$\|f_\theta(\mathbf{x}_t, t) - f_\theta(\mathbf{y}_t, t)\|_2 \le L\|\mathbf{x}_t - \mathbf{y}_t\|_2.$$

Additionally, assume that for each step $n \in \{1, 2, \ldots, N-1\}$, the ODE solver called at t_n has a local error bounded by $O((t_{n+1} - t_n)^{p+1})$ for some $p \ge 1$.

If, additionally, $\mathcal{L}_{CD}(\theta, \theta; u) = 0$, then:

$$\sup_{n, \mathbf{x}_{t_n}} \|f_\theta(\mathbf{x}_{t_n}, t_n) - f(\mathbf{x}_{t_n}, t_n; u)\|_2 = O((\Delta t)^p),$$

where $\Delta t := \max_{n \in \{1,2,\ldots,N-1\}} |t_{n+1} - t_n|$.

Proof. The proof is similar to the one presented by Song et al. (2023), with the key difference that we must account for the global integration error introduced by the ODE solver.

If the ODE solver, when called at t_{n+1}, has a local error uniformly bounded by $O((t_n - t_{n-1})^{p+1})$, then the cumulative error across all steps is approximately the sum of $n + 1$ local errors and is bounded by $O((\Delta t)^p)$.

We are interested in e_n, the error between the learned consistency function and the consistency function of the PF ODE defined by the control u at $\mathbf{x}_{t_n} \sim p_{t_n}(\mathbf{x}_{t_n})$,

$$e_n := f_\theta(\mathbf{x}_{t_n}, t_n) - f(\mathbf{x}_{t_n}, t_n; u).$$

If $\mathcal{L}(\theta, \theta; u) = 0$, we deduce that

$$\lambda(t_n)d(f_\theta(\hat{\mathbf{x}}_{t_{n+1}}, t_{n+1}), f_\theta(\hat{\mathbf{x}}_{t_n}, t_n)) = 0.$$

Since $\lambda(t_n) > 0$, this implies:

$$f_\theta(\hat{\mathbf{x}}_{t_{n+1}}, t_{n+1}) = f_\theta(\hat{\mathbf{x}}_{t_n}, t_n). \tag{11}$$

We can derive a recurrence relation for e_n:

$$\begin{aligned}
e_n &\overset{(i)}{=} f_\theta(\mathbf{x}_{t_n}, t_n) - f_\theta(\hat{\mathbf{x}}_{t_n}, t_n) + f_\theta(\hat{\mathbf{x}}_{t_n}, t_n) - f(\mathbf{x}_{t_{n+1}}, t_{n+1}; u) \\
&\overset{(ii)}{=} f_\theta(\mathbf{x}_{t_n}, t_n) - f_\theta(\hat{\mathbf{x}}_{t_n}, t_n) + f_\theta(\hat{\mathbf{x}}_{t_{n+1}}, t_{n+1}) - f(\mathbf{x}_{t_{n+1}}, t_{n+1}; u) \\
&= f_\theta(\mathbf{x}_{t_n}, t_n) - f_\theta(\hat{\mathbf{x}}_{t_n}, t_n) + f_\theta(\hat{\mathbf{x}}_{t_{n+1}}, t_{n+1}) - f_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}) \\
&\quad + f_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}) - f(\mathbf{x}_{t_{n+1}}, t_{n+1}; u) \\
&= f_\theta(\mathbf{x}_{t_n}, t_n) - f_\theta(\hat{\mathbf{x}}_{t_n}, t_n) + f_\theta(\hat{\mathbf{x}}_{t_{n+1}}, t_{n+1}) - f_\theta(\mathbf{x}_{t_{n+1}}, t_{n+1}) + e_{n+1} \\
&\cdots \\
&\overset{(iii)}{=} f_\theta(\mathbf{x}_{t_n}, t_n) - f_\theta(\hat{\mathbf{x}}_{t_n}, t_n) + f_\theta(\mathbf{x}_T, T) - f_\theta(\hat{\mathbf{x}}_T, T) + e_T.
\end{aligned}$$

Here, step (i) follows from the definition of the consistency function, step (ii) is due to Eq. (11), and step (iii) leverages the telescoping nature of the sum.

Furthermore, since f_θ is parameterized such that $f_\theta(\mathbf{x}_T, T) = \mathbf{x}_T$, we have

$$\begin{aligned}
e_T &= f_\theta(\mathbf{x}_T, T) - f(\mathbf{x}_T, T; u) \\
&= \mathbf{x}_T - \mathbf{x}_T \\
&= 0.
\end{aligned}$$

Finally, given that f_θ is Lipschitz and considering the bound on the global error of the ODE solver:

$$\|e_n\|_2 \le \|e_T\|_2 + L\|\mathbf{x}_{t_n} - \hat{\mathbf{x}}_{t_n}\|_2 + L\|\mathbf{x}_T - \hat{\mathbf{x}}_T\|_2 = O((\Delta t)^p).$$

$\square$

C EXPERIMENTAL DETAILS

C.1 TARGET DISTRIBUTIONS

GMM. Here we discuss the parameterization for the Gaussian mixture model with well separated modes. We follow the same setting as Zhang & Chen (2022); Berner et al. (2024), defining the target distribution as follows:

$$\rho(\mathbf{x}) = \sum_{m=1}^{M} \alpha_m \mathcal{N}(\mathbf{x}; \mu_m, \Sigma_m)$$

Following their pararameterization, we set $M = 9$, $\sigma_m = .3I$, and $(\mu)_{m=1}^{M} = \{-5, -, 5\} \times \{-5, 0, 5\} \subset \mathbb{R}^2$.

Image. We use a normalized grayscale image to create a two-dimensional probability density, following the setup from Wu et al. (2020).

Funnel. Following the methodology of Berner et al. (2024), we use the funnel distribution introduced from Neal (2003). The distribution is defined as follows:

$$\rho(\mathbf{x}) = \mathcal{N}(x_1; 0, v^2) \prod_{i=1}^{d} \mathcal{N}(x_i; 0, e^{x_1})$$

We set $d = 10, v = 3$.

We include this benchmark as this is a canonical distribution used for comparing MCMC methods and has been used extensively within the growing field of learned diffusion samplers (Berner et al., 2024; Zhang & Chen, 2022; Vargas et al., 2023; Richter & Berner, 2024).

Many-Well. We use the many-well target distribution following the methodology of Berner et al. (2024):

$$\rho(\mathbf{x}) = \exp\left(-\sum_{i=1}^{m}(x_i^2 - \delta) - \frac{1}{2}\sum_{i=m+1}^{d} x_i^2\right).$$

For the target distribution labeled as MW-54, we set $d = 5$, $m = 5$, and $\delta = 4$; for the target distribution labeled as MW-52, we set $d = 50, m = 5, \delta = 2$.

Log Cox Gaussian Process (LGCP). The log cox Gaussian process is a popular target distribution for benchmarking sampling methods due to its complexity and high-dimensionality. As discussed in Zhang & Chen (2022); Chen et al. (2025), the LGCP distribution is defined as follows:

$$\rho(x) = \mathcal{N}(x; \mu, \Sigma) \prod_{i=1}^{d} \exp\left(x_i y_i - \frac{\exp(x_i)}{d}\right).$$

Here, y is a given dataset, and μ, Σ are mean and covariance for some given prior. We follow the methodology of Zhang & Chen (2022); Arbel et al. (2021) for both the dataset and prior distribution.

C.2 TRAINING DETAILS

For GMM, image, funnel, and MW54, we train all diffusion samplers until convergence or for 30,000 training iterations. For MW52d, we train all samplers for 10,000 training iterations. For LGCP, we train all samplers for 5,000 training iterations.

For a complete specification of sampler details, see Table 3. For details on the global configurations used across all samplers, see Table 4.

SCDS	
Terminal Time	1
SDE	VP SDE
Terminal Time	1
Time Schedule	Linear
Initial Distribution	$\mathcal{N}(0, I)$ with Truncation Quartile of $1e - 4$
Loss Function	Log-Variance, Time Reversal (Berner et al., 2024; Richter & Berner, 2024), Self-Consistency

CDDS	
Pretrained Generative Ctrl	DIS
Consistency Model Train Timesteps	18
Loss Function	Equation equation 6

DIS (Berner et al., 2024)	
SDE	VP SDE
Loss Function	Log Variance, Time Reversal
Terminal Time	1
Time Schedule	Linear
Initial Distribution	$\mathcal{N}(0, I)$ with Truncation Quartile of $1e - 4$

PIS (Zhang & Chen, 2022)	
SDE	VE SDE
Loss Function	Log Variance (Richter & Berner, 2024)
Terminal Time	1
Time Schedule	Linear
Initial Distribution	Dirac-Delta

DDS (Vargas et al., 2023)	
SDE	VP SDE
Loss Function	Log Variance
SDE	Exponential SDE
Time Schedule	Cosine
Terminal T	12.8
Δt	.1
Initial Distribution	$\mathcal{N}(0, I)$ with Truncation Quartile of $1e - 4$

Table 3: Diffusion Sampler Configurations

| Optimizer Settings	
Optimizer	Adam
Learning Rate	.005
Weight Decay	$1e-7$
Gradient Clipping	1
β_1, β_2	.9, .999
Training Settings	
Total Iterations	GMM, Image, Funnel, MW54=30,000; MW52=10,000; LGCP=5,000
Train Time Steps	128
Batch Size	2048
Model Settings	
Number of Layers	4
Channels	64
Time Conditioning	Fourier Time Embeddings Tancik et al. (2020)
Activation	GeLU
Evaluation Settings	
Batch Size	10000
Weight Decay	$1e-7$

Table 4: Global Configurations
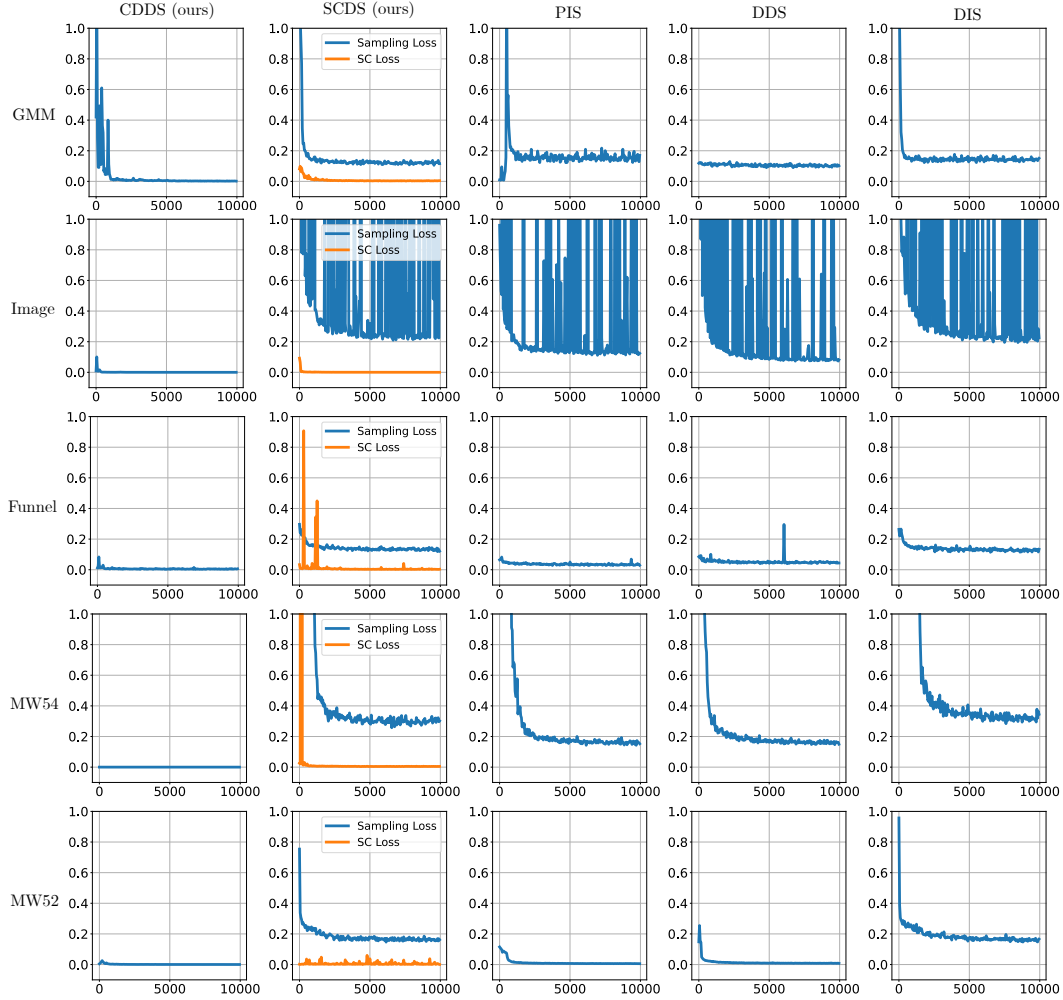
Figure 3: Loss curves for the samplers studied in this paper. SCDS and CDDS exhibit stable learning across most settings, except for the image target distribution, where all samplers—except CDDS—show instability. Notably, the self-consistency loss and the sampling loss remain relatively independent.