NO TRICK, NO TREAT: PURSUITS AND CHALLENGES TOWARDS SIMULATION-FREE TRAINING OF NEURAL SAMPLERS

Anonymous authors
Paper under double-blind review

ABSTRACT

We consider the *sampling* problem, where the aim is to draw samples from a distribution whose density is known only up to a normalization constant. Recent breakthroughs in generative modeling to approximate a high-dimensional data distribution have sparked significant interest in developing neural network–based methods for this challenging problem. However, neural samplers typically incur heavy computational overhead due to simulating trajectories during training. This motivates the pursuit of *simulation-free* training procedures of neural samplers. In this work, we propose an elegant modification to previous methods, which allows simulation-free training with the help of a time-dependent normalizing flow. However, it ultimately suffers from severe mode collapse. On closer inspection, we find that nearly all successful neural samplers rely on Langevin preconditioning to avoid mode collapsing. We systematically analyze several popular methods with various objective functions and demonstrate that, in the absence of Langevin preconditioning, most of them fail to adequately cover even a simple target. Finally, we draw attention to a strong baseline by combining the state-of-the-art MCMC method, Parallel Tempering (PT), with an additional generative model to shed light on future explorations of neural samplers.

1 INTRODUCTION

Sampling is a fundamental task in statistics, with broad applications in Bayesian inference, rare event sampling, and molecular simulation (Box & Tiao, 2011; Tuckerman, 2023; Dellago et al., 2002; Du et al., 2024). Consider a target distribution with the following density function:

$$p_{\text{target}}(x) = \frac{\tilde{p}_{\text{target}}(x)}{Z}, \quad Z = \int_{\Omega} \tilde{p}_{\text{target}}(x)dx, \tag{1}$$

where $\tilde{p}_{\text{target}}(x)$ is the unnormalized density which we can evaluate for a given x, and Z is an unknown normalization factor. We aim to generate samples following p_{target}. These samples can be used to estimate the normalization factor or the expectation over some test functions.

A "standard" solution to this problem is Markov chain Monte Carlo (MCMC), which runs a Markov chain whose invariant density is p_{target}. Building on top of MCMC, various advanced sampling techniques have been developed, with the most efficient methods including Parallel Tempering (PT) (Swendsen & Wang, 1986b; Earl & Deem, 2005), Annealed Importance Sampling (AIS) (Neal, 2001), and Sequential Monte Carlo (SMC) (Doucet et al., 2001). However, MCMC-based approaches typically suffer from slow mixing time and dependency between samples.

A growing trend of research directions therefore focus on the learned neural sampler, e.g., (Noé et al., 2019), where we train a neural network to amortize the sampling procedure. Initial attempts studied normalizing flows (NFs) and used them as proposals for importance sampling (IS) (Noé et al., 2019; Midgley et al., 2023). Later, diffusion and control-based samplers gained notable attention (Zhang & Chen, 2022; Doucet et al., 2022; Vargas et al., 2023; Berner et al., 2024; Vargas et al., 2024; Albergo & Vanden-Eijnden, 2024) due to their success in generative modeling (Ho et al., 2020; Song et al., 2021; Karras et al., 2022). These methods start with an easy-to-sample distribution (e.g., Gaussian) and evolve them through a stochastic differential equation (SDE) or ordinary differential equation (ODE). However, despite significant progress, these approaches typically require

simulating the entire trajectory to evaluate the training objective. For instance, the most common objective - the reverse KL divergence between the model path measure and the target path measure - generally necessitates simulating the full trajectory for every sample and backpropagating through it. This leads to substantial memory consumption and slows down the training process.

To this end, various objectives have been proposed to reduce computational costs. Some off-policy objectives enable detaching the gradient from the simulation (Richter & Berner, 2024), while others involve simulating only a partial path (Zhang et al., 2024). The ultimate goal, however, is to design a sampler and training objective that can be optimized without any trajectory simulation following lessons learned from diffusion and flow matching models (Ho et al., 2020; Song et al., 2021; Lipman et al., 2023). While appealing, we argue that most current approaches are not well-suited for such a design. This obstacle stems not only from how to modify the objective formulation for simulation-free evaluation but also from these approaches' reliance on tricks in network parameterization and sampling procedures that are not compatible with simulation-free training - most notably, the Langevin preconditioning, first proposed by Zhang & Chen (2022). Through a simple example, we demonstrate that even with the same objective and a mode covering initialization, simulation-free training leads to significant mode collapse. We attribute this failure to the absence of the Langevin preconditioning in the simulation-free training pipeline. To further support this claim, we provide ablation studies, showing that most current approaches struggle without the Langevin preconditioning. This observation highlights critical caveats and considerations that must be addressed in future work aimed at developing training-free objectives and pipelines.

Running simulations with the Langevin preconditioning also poses a new challenge: simulation during training greatly increases the number of evaluations of the target density, which can be prohibitively expensive in some applications. Consequently, it remains unclear whether these approaches are efficient compared to directly running MCMC and fitting a diffusion sampler post-hoc. To investigate this, we compare the samplers with a state-of-the-art MCMC method, Parallel Tempering (PT, Swendsen & Wang, 1986a; Earl & Deem, 2005). We find that PT serves as a remarkably strong baseline that should not be overlooked.

In summary, our main contributions are as follows: (1) We provide a systematic review of current samplers, focusing on classifying different approaches by their underlying process and objectives. (2) We propose a simple direction for achieving it using Normalizing Flows. Unfortunately, this attempt does not perform as desired, which we attribute to the absence of Langevin preconditioning widely applied in other neural samplers. (3) We investigate the influence of Langevin preconditioning. Our findings reveal that most approaches fail when the sampler is not parameterized with the gradient of the target density. This indicates critical caveats and considerations that should be addressed in developing simulation-free approaches. (4) Finally, we compare several diffusion neural samplers with PT, and find that they lag significantly behind the results obtained from running traditional MCMC methods and fitting a diffusion model post hoc. This highlights key challenges and critical considerations for enhancing the practicality of neural samplers in future work.

2 Review of Diffusion and Controlled Samplers

Before discussing the potential design of a simulation-free training approach, we first present a systematic review of current diffusion and controlled-based samplers in this section. Despite the abundance of existing approaches, most samplers can be broadly categorized based on their sampling processes and training objectives:

1. **Sampling processes**. We can write the sampling process as follows:

$$dX_t = \left[\mu_t(X_t) + \sigma_t^2 b_t(X_t)\right] dt + \sigma_t \sqrt{2} dW_t, \quad X_0 \sim p_{\text{prior}}, \tag{2}$$

fixing or learning $\{\mu_t, \sigma_t, b_t, p_{\text{prior}}\}$ results in different sampling strategies. Broadly, there are three main types of processes:

- *time-reversal sampler*: the first involves training Equation (2) to approximate the time-reversal of a target process that begins with the target p_{target} and evolves toward a tractable distribution such as p_{prior}. The target process is typically designed with a tractable drift term to ensure that its terminal density (approximately) converges to p_{prior}, with common choices including variance-preserving (VP) and variance-exploding (VE) SDEs and pinned Brownian motion (PBM). This category includes methods like PIS (Zhang & Chen, 2022; Vargas et al., 2021), DDS (Vargas et al., 2023), DIS (Berner et al., 2024), and iDEM (Akhound-Sadegh et al., 2024).

Table 1: Properties of different sampling processes.

Underlying process	**Properties**		
	non-ergodicity	arbitrary p_{prior}	no mode switching
Reversal of VP/VE SDE	✗	✗	✓
Reversal of PBM	✓	✗	✓
Escorted Transport (geom. interpolate)	✓	✓	✗

- *escorted transport sampler*: the second trains Equation (2) to transport between a sequence of prescribed marginal densities π_t, typically defined by interpolation between p_{prior} and p_{target}, with $\pi_0 = p_{\text{prior}}$ and $\pi_T = p_{\text{target}}$. Representative methods include Escorted Jarzynski (Vaikuntanathan & Jarzynski, 2011), CMCD (Vargas et al., 2024), NETS / PINN-based transport (Máté & Fleuret, 2023; Albergo & Vanden-Eijnden, 2024), LFTS (Tian et al., 2024), etc.
- *annealed variance reduction sampler*: Similar to the escorted transport sampler, these approaches prespecify an annealed target π_t and set $b_t = 0$ and $\mu_t = \nabla \ln \pi_t$ just like the proposal in AIS (Neal, 2001; Jarzynski, 1997), the forward process remains fixed so no guidance/escorting is learned. However, one approximates the reversal of this forward proposal so that the Radon-Nikodym derivative (RND) between the time-reversal and the forward proposal has a low variance, allowing a more efficient importance sampling. This category includes methods like AIS (Neal, 2001; Jarzynski, 1997), MCD (Doucet et al., 2022; Zhang, 2021; Hartmann et al., 2019), LDVI (Geffner & Domke, 2023), among others.

We compare the properties of different underlying processes in Table 1, including ergodicity (i.e., whether the sampler can mix within a finite number of steps (Albergo et al., 2023; Huang et al., 2021; Zhang & Chen, 2022; Vargas et al., 2021; Grenioux et al., 2024)), flexibility on the choice of prior, and the "smoothness" (Chemseddine et al., 2024; Woodard et al., 2009; Tawn et al., 2020; Syed et al., 2022; Phillips et al., 2024) of the induced flow (i.e. the mass teleportation problem, also known as mode switching).

2. **Training objectives**. There are mainly two families of objectives:

- *path measure alignment*: the first one aligns the path measure induced by the sampling process, i.e., the SDE starting from p_{prior}, with another process starting from the target distribution p_{target} and traversing in reverse. Common objectives include KL divergence (Zhang & Chen, 2022; Vargas et al., 2021; 2023; Doucet et al., 2022; Lahlou et al., 2023; Berner et al., 2024; Vargas et al., 2024), log-variance divergence (Richter & Berner, 2024), the (sub-)trajectory balance objective (Zhang et al., 2024), and detailed balance objective (Bengio et al., 2021).
- *marginal alignment*: this approach aims to align the drift term or vector field of the sampling process with a prescribed target, ensuring that the marginal distributions of the generated samples closely follow the desired trajectory at each time step. Common objectives in this category include the physics-informed neural network (PINN) loss (Sun et al., 2024; Albergo & Vanden-Eijnden, 2024), action matching loss (Albergo & Vanden-Eijnden, 2024), and score matching with importance sampling (Akhound-Sadegh et al., 2024).

We compare the properties of different objectives in Table 2. Specifically, we assess whether they support off-policy training, can be computed without simulation, require the costly calculation of network divergence, and ensure unbiasedness.

We note that the *simulation-free training* can relate to several concepts in the neural sampler literature: (1) training without using MCMC, (2) detaching gradients on samples when evaluating trajectory-based objectives, and (3) evaluating objectives at any time step without simulating the trajectory. In this paper, we formally define simulation-free training as training with an objective that can be evaluated without simulating any ODE or SDE, aligning with the principles of diffusion and flow matching methods.

Combining different underlying processes and objectives, we will recover many common neural samplers. In the following, we briefly explain their design and categorize them in Table 3. We include more details in Appendix C.

(1) Path Integral Sampler (PIS, Zhang & Chen, 2022) and concurrently (NSFS, Vargas et al., 2021): PIS fixes $p_{\text{prior}} = \delta_0, \sigma_t = 1/\sqrt{2}$ and learns a network $f_\theta(\cdot) = \mu_t(\cdot) + \sigma_t^2 b_t(\cdot)$ so that Equa-

Table 2: Properties of different objectives. *KL divergence does not support simulation-free training in general. However, it can be calculated without simulation for some special cases. We will provide an example later in Section 3. **KL divergence and log-variance divergence typically do not require computing the divergence. However, Richter & Berner (2024) proposed objectives for neural samplers based on the general Schrödinger Bridge that requires computing this divergence.

Objective	**Properties**			
	off-policy	sim-free	div-free	unbiased
KL	✗	✗(✓*)	✓(✗**)	✓
LV	✓	✗	✓(✗**)	✓
TB/STB	✓	✗	✓	✓
DB	✓	✗	✓	✓
PINN	✓	✓	✗	✓
AM	✓	✗	✓	✓
SM w. IS	✓	✓	✓	✗

tion (2) approximate the time-reversal of the following SDE (Pinned Brownian Motion):

$$dY_t = -\frac{Y_t}{T-t}dt + dW_t, \quad Y_0 \sim p_{\text{target}}. \tag{3}$$

We define Equation (3) as the time-reversal of Equation (2) when $Y_t \sim X_{T-t}$. The network is learned by matching the reverse KL (Zhang & Chen, 2022; Vargas et al., 2021) or log-variance divergence (Richter & Berner, 2024) between the sampling and the target process.

Diffusion generative flow samplers (DFGS, Zhang et al., 2024) time-reversal the same pinned Brownian motion but with a new introduction of local objectives including detailed balance and (sub-)trajectory balance which has been shown equivalent to the log-variance objective with a learned baseline rather than a Monte Carlo (MC) estimator (Nüsken & Richter, 2021).

(2) Denoising Diffusion Sampler (DDS, Vargas et al., 2023) and time-reversed Diffusion Sampler (DIS, Berner et al., 2024): both DDS and DIS fix $\mu_t(X_t, t) = \beta_{T-t}X_t, \sigma_t = v\sqrt{\beta_{T-t}}, p_{\text{prior}} = \mathcal{N}(0, v^2 I)$, and learn a network $f_\theta(\cdot, t) = b_t(\cdot, t)/2$ so that Equation (2) approximates the time-reversal of the VP-SDE:

$$dY_t = -\beta_t Y_t dt + v\sqrt{2\beta_t}dW_t, \quad Y_0 \sim p_{\text{target}}. \tag{4}$$

In an optimal solution, f_θ will approximate the score $f_\theta(\cdot, t) \approx \nabla \log p_{T-t}(\cdot)$, where $p_t(X) = \int \mathcal{N}(X|\sqrt{1-\lambda_t}Y, v^2\lambda_t I)p_{\text{target}}(Y)dY$ and $\lambda_t = 1 - \exp(-2\int_0^t \beta_s ds)$. Similar to PIS, the network can be trained either with reverse KL divergence or log-variance divergence.

(3) Iterated Denoising Energy Matching (iDEM, Akhound-Sadegh et al., 2024): iDEM fixes $\mu_t(X_t, t) = 0, p_{\text{prior}} = \mathcal{N}(0, T^2 I)$, and learns a network $f_\theta(\cdot, t) = b_t(\cdot, t)/2$ to approximate the score $f_\theta(\cdot, t) \approx \nabla \log p_{T-t}(\cdot)$, where $\log p_{T-t}$ is estimated by target score identity (TSI, De Bortoli et al., 2024) with a self-normalized importance sampler:

$$\nabla \log p_{T-t}(X_t) \approx \sum_n \frac{\tilde{p}_{\text{target}}(X_T^{(n)})}{\sum_m \tilde{p}_{\text{target}}(X_T^{(m)})} \nabla \log \tilde{p}_{\text{target}}(X_T^{(n)}), \quad X_T^{(n)} \sim q_{T|t}(X_T|X_t), \tag{5}$$

$q_{T|t}(X_T|X_t)$ is the importance sampling proposal chosen as $q_{T|t}(X_T|X_t) \propto p_{t|t}(X_t|X_T)$. In an optimal solution, the sampling process approximates the time-reversal of a VE-SDE:

$$dY_t = \sqrt{2t}dW_t, \quad Y_0 \sim p_{\text{target}}. \tag{6}$$

One can re-interpret the estimator regressed in iDEM in terms of the optimal drift solving a stochastic control problem (Huang et al., 2021). The optimal control $f^*_{\sigma_{\text{init}}}$ can be expressed in terms of the score (e.g. See Remark 3.5 in Reu et al. (2024)), for any $\sigma_{\text{init}} > 0$:

$$f^*_{\sigma_{\text{init}}}(X_t, t) = -\nabla \log \phi_{T-t}(X_t) = -\frac{X_t}{T - t + \sigma^2_{\text{init}}} + \nabla \log p_{T-t}(X_t), \tag{7}$$

where $\phi_t(X_t)$ is the value function. It can be expressed as a conditional expectation via the Feynman-Kac formula with Hopf-Cole transform (Hopf, 1950; Cole, 1951; Fleming, 1989):

$$\phi_t(X_t) = \mathbb{E}_{X_T \sim q_{T|t}(X_T|X_t)}\left[\frac{\tilde{p}_{\text{target}}}{\mathcal{N}(0, T + \sigma^2_{\text{init}})}(X_T)\right]. \tag{8}$$

Table 3: We obtain common neural samplers by combining different underlying processes and objectives. DDS: Vargas et al. (2023); DIS: Berner et al. (2024); DDS/DIS/PIS-LV: (Richter & Berner, 2024); CMCD: (Vargas et al., 2024); NETS: Albergo & Vanden-Eijnden (2024); PINN-based: Sun et al. (2024); RDMC: Huang et al. (2023); iDEM: Akhound-Sadegh et al. (2024); SFS: Huang et al. (2021); LFIS: Tian et al. (2024); GFN: Zhang et al. (2024). *RDMC and SFS only estimate the score/optimal control function by importance sampling, and do not evolve network training.

	KL	**LV**	**TB/STB**	**DB**	**PINN**	**AM**	**Score Estimation**
Reversal of VP/VE SDE	DDS, DIS	DDS-LV, DIS-LV			PINN-based		RDMC*, iDEM
Reversal of PBM	PIS	PIS-LV	DGFS	DGFS			SFS*
Escorted Transport	CMCD	CMCD			PINN-based, NETS, LFIS	NETS	

Note the MC Estimator of $\nabla \log \phi_{T-t}(X_t)$ (e.g. Equation 8) was used in Schrödinger-Föllmer Sampler (SFS, Huang et al., 2021) to sample from time-reversal of pinned Brownian Motion, yielding an estimator akin to the one used in iDEM.

(4) Monte Carlo Diffusion (MCD, Doucet et al., 2022): unlike other neural samplers, MCD's sampling process is fixed as $\mu_t = 0, \sigma_t = 1, b_t(X_t, t) = \nabla \log \pi_t(X_t)$, where π_t is the geometric interpolation between target and prior, i.e., $\pi_t(X_t) = p^{\beta_t}_{\text{target}}(X_t) p^{1-\beta_t}_{\text{prior}}(X_t)$. It can be viewed as sampling with AIS using ULA as the kernel. Note, that this transport is non-equilibrium, as the density of X_t is not necessary $\pi_t(X_t)$. Therefore, MCD trains a network to approximate the time-reversal of the forward process and perform importance sampling (more precisely, AIS) to correct the bias of the non-equilibrium forward process.

(5) Controlled Monte Carlo Diffusion (CMCD, Vargas et al., 2024) and Non-Equilibrium Transport Sampler (NETS, Máté & Fleuret, 2023; Albergo & Vanden-Eijnden, 2024): Similar to MCD, CMCD and NETS also set $b_t(X_t, t) = \nabla \log \pi_t(X_t)$ and π_t is the interpolation between target and prior[1]. Different from MCD where the sampling process is fixed, CMCD and NETS learn $f_\theta(\cdot, t) = \mu_t(\cdot, t)$ so that the marginal density of samples X_t simulated by Equation (2) will approximate π_t. As a special case, Liouville Flow Importance Sampler (LFIS, Tian et al., 2024) fixes $\sigma_t = 0$ and learns an ODE to transport between π_t.

3 Simulation-free Training with Normalizing Flow Induced SDEs

In this section, we propose a potential design for simulation-free training of DDS and CMCD using normalizing flows (NF)[2]. Consider a time-dependent normalizing flow defined as $F_\theta : \mathcal{X} \times [0, T] \to \mathcal{X}$. We denote the density of the samples drawn from the normalizing flow as $q_\theta(X_t, t)$. A key property of NFs that enables simulation-free training is their ability to generate samples $X_t \sim q_\theta(X_t, t)$ through two distinct approaches (Bartosh et al., 2024):

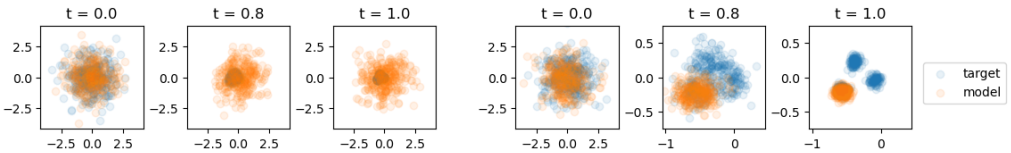
1. Drawing from the base distribution $X_{\text{base}} \sim p_{\text{base}}$ and transforming it via $F_\theta(X_{\text{base}}, t)$;
2. Drawing an initial sample $X_{\text{base}} \sim p_{\text{base}}$, $X_0 = F_\theta(X_{\text{base}}, 0)$ and evolving through an ODE: $dX_t = \partial_t F_\theta(X_{\text{base}}, t)dt = (\partial_t F_\theta(X_{\text{base}}, t)|_{X_{\text{base}} = F_\theta^{-1}(X_t, t)})dt$. For simplicity, we write $\tilde{F}_\theta(X_t, t) = \partial_t F_\theta(X_{\text{base}}, t)|_{X_{\text{base}} = F_\theta^{-1}(X_t, t)}$. Additionally, the following SDE will have the same marginal density as the ODE for any $\sigma \geq 0$:

$$dX_t = \left(\tilde{F}_\theta(X_t, t) + \sigma_t^2 \nabla \log q_\theta(X_t, t) \right) dt + \sigma_t \sqrt{2} dW_t. \tag{9}$$

The first approach allows us to directly generate samples along the trajectory without simulation, while the second approach allows the use of the same objective as previously described in control-based samplers. In the following, we introduce NF-DDS, leveraging normalizing flows to achieve a

[1] CMCD defines π_t with geometric interpolation between target and prior $\pi_t(X_t) = p^{\beta_t}_{\text{target}}(X_t) p^{1-\beta_t}_{\text{prior}}(X_t)$. In contrast, NETS defines π_t differently depending on the target distribution. For example, with a GMM target, π_t is constructed as a GMM whose components' means and variances are linearly interpolated between the target mixture components and a Gaussian around 0. We will denote this as mode interpolation.

[2] We use NF to refer to an invertible network rather than continuous normalizing flows (Chen et al., 2018).



(a) Initialization of NF-DDS, samples generated at different time steps $0, 0.8, 1.0$. As we can see, the initialization already covers all modes.

(b) NF-DDS after training with Equation (13), samples generated at different time steps $0, 0.8, 1.0$. Unlike DDS, NF-DDS fails to capture all modes.

simulation-free training objective. In Appendix E, we present an alternative approach, NF-CMCD, which coincides with matching the reverse Fisher divergence between marginals in all time steps.

NF-DDS: Recall that in DDS, we match the sampling process in Equation (9) with the time-reversal of a VP-SDE starting from the target density:

$$dY_t = -\beta_t Y_t dt + v\sqrt{2\beta_t} dW_t, \quad Y_0 \sim p_{\text{target}}. \tag{10}$$

To have a bounded RND between the target path measure and Equation (9), we set $\sigma_t = v\sqrt{\beta_{T-t}}$. We rewrite the sampling process for easy reference:

$$dX_t = \left(\tilde{F}_\theta(X_t, t) + v^2 \beta_{T-t} \nabla \log q_\theta(X_t, t)\right) dt + v\sqrt{2\beta_{T-t}} dW_t. \tag{11}$$

By Nelson's condition (Nelson, 1967; Anderson, 1982), we can write its time-reversal as

$$dY_t = -\left(\tilde{F}_\theta(Y_t, T-t) - v^2 \beta_t \nabla \log q_\theta(Y_t, T-t)\right) dt + v\sqrt{2\beta_t} dW_t, Y_0 \sim q_\theta(Y_0, T). \tag{12}$$

By Girsanov theorem, the KL divergence $D_{\text{KL}}[\mathbb{Q}||\mathbb{P}]$ between the path measure induced by Equation (12) (denoted as $\mathbb{Q}$) and Equation (10) (as $\mathbb{P}$) is tractable (derivation details in Appendix D):

$$\int_0^T \frac{1}{4v^2\beta_{T-t}} \mathbb{E}_{q_\theta(Y,t)} \|\tilde{F}_\theta(Y,t) - v^2\beta_{T-t}\nabla \log q_\theta(Y,t) - \beta_{T-t} Y\|^2 dt + D_{\text{KL}}[q_\theta(\cdot, T)||p_{\text{target}}]. \tag{13}$$

Failure of NF-DDS: Although NF-DDS enables simulation-free training for DDS, it struggles to perform well even on simple tasks. We evaluate NF-DDS by training it on a 2D 3-mode Gaussian Mixture target distribution. Figures 1a and 1b illustrate the initialization and the outcomes after training. Despite starting with an initialization that covers all modes, and being optimized using the same objective as DDS, NF-DDS fails to achieve satisfactory results.

What is the difference between DDS and NF-DDS leading to this performance discrepancy? Excluding the influence of objectives, the only difference left is the model. Specifically, DDS adopts the network proposed by PIS (Zhang & Chen, 2022):

$$f_\theta(\cdot, t) = \text{NN}_{1,\theta}(\cdot, t) + \text{NN}_{2,\theta}(t) \circ \nabla \log p_{\text{target}}(\cdot), \tag{14}$$

and initializes $\text{NN}_{1,\theta} \approx 0$ and $\text{NN}_{2,\theta} = 1$. In the early stages of training, DDS simulation closely resembles running MCMC with Langevin dynamics. In fact, nearly all algorithms discussed in Section 2 incorporate a similar term, either explicitly or implicitly. If simulating these Langevin terms plays a crucial role, then modifying current algorithms to achieve simulation-free training may not be straightforward or even infeasible. Therefore, in the next section, we provide ablation studies on the influence of the Langevin term, which we denote as *Langevin preconditioning*, in both time-reversal sampler and escorted transport sampler, trained with different objectives.

4 Ablation on Langevin Preconditioning and Its Implications

In this section, we ablate the effectiveness of the Langevin preconditioning on examples of different neural samplers. For the time-reversal sampler, we take DDS as an example, while for the escorted transport sampler, we take CMCD as an example. We will explore objectives including reverse KL, Log-var divergence, trajectory balance, and PINN.

First, we discuss how to remove Langevin preconditioning in different samplers:

Table 4: Sample quality of time-reversal sampler and escorted transport sampler trained with different objectives. We compare their performances both with and without the Langevin preconditioning. We measure MMD, EUBO and ELBO. MMD can have a comprehensive reflection on the sample quality, and the difference between EUBO and ELBO measures the mode coverage: large EUBO indicates mode collapsing. As some methods diverge in the end, we report the results with early stopping, according to ELBO. N/A denotes unstable training, and no reasonable result is obtained.

Obj.	**DDS**					**CMCD**			
	w. LG	w/o LG	w. $\log p_{\text{target}}$	distil init.	w/o LG + init.	w. LG	w/o LG	distil init.	w/o LG + init.
					MMD (↓)				
rKL	0.074	1.497	4.260		0.333	0.075	4.011		1.827
LV	0.064	1.938	1.995	0.121	0.014	0.017	N/A	0.079	0.036
TB	0.054	4.413	4.550		0.015	0.035	N/A		0.130
					ELBO (↑) /EUBO (↓)				
rKL	-0.45/0.49	-1.93/28.52	-2.36/35.02		-1.14/3.03	-0.40/0.45	-4.45/193.06		$-3.28/3\times10^5$
LV	-0.90/0.77	-2.07/16.26	-1.96/17.19	-0.88/0.64	-0.53/0.44	-0.28/0.33	N/A	-0.89/0.82	-0.53/0.77
TB	-1.73/1.36	-2.62/23.00	-2.61/28.75		-0.46/0.45	-0.52/0.77	N/A		-0.77/1.20

- *DDS without Langevin Preconditioning*. DDS's Langevin preconditioning occurs in its network parameterization. Therefore, to eliminate the help of Langevin during simulation in the training process, we can simply replace the network in Equation (14) by a standard MLP. To ensure the model capacity, we increase the MLP size to 5 layers with 256 hidden units.
- *CMCD without Langevin Preconditioning*. Unlike DDS, Langevin preconditioning in CMCD naturally emerges from its formulation. Specifically, CMCD defines the drift terms for the sampling and "target" processes as $f_\theta(X_t, t) + \sigma_t^2 \nabla \log \pi_t(X_t)$ and $-(f_\theta(Y_t, T-t) - \sigma_t^2 \nabla \log \pi_{T-t}(Y_t))$ respectively. By aligning their path measures, the marginal density of the sampling process at time t is ensured to match π_t in accordance with Nelson's condition (Nelson, 1967). In order to eliminate the Langevin preconditioning $\nabla \log \pi_t(X_t)$ during simulation in training, we redefine the sampling and "target" processes as $f_\theta(X_t, t)$ and $-(f_\theta(Y_t, T-t) - 2\sigma_t^2 \nabla \log \pi_{T-t}(Y_t))$. Aligning their path measures still ensures that the marginal density of the sampling process at time t matches π_t, while the training simulation does not rely on the help of Langevin preconditioning.
- *PINN without Langevin Preconditioning*. In CMCD/NETS, sampling process is defined as $dX_t = \big(f_\theta(X_t, t) + \sigma_t^2 \nabla \log \pi_t(X_t)\big)\, dt + \sigma_t \sqrt{2} dW_t$, and the objective is independent of the value of σ_t. Therefore, we simply set $\sigma_t = 0$ during training to eliminate the Langevin preconditioning.

Additionally, we investigate the performance of DDS and CMCD when the initialization is close to optimal. To achieve this, we first train DDS and CMCD with Langevin preconditioning until convergence. Then, we use a new network without Langevin preconditioning to distill the teacher output with Langevin preconditioning at each time step using an L_2 loss. After distillation, we fine-tune the student network using different objectives. This allows us to examine whether Langevin preconditioning primarily aids in localizing the model in the early training stage or also contributes to stabilizing the results in the end of training.

For DDS, we also test the results using a network conditioned on the target density instead of the target score: $f_\theta(X, t) = \text{NN}_\theta(X, \log \tilde{p}_{\text{target}}(X), t)$. This allows us to verify whether neural samplers require an explicit score term to ensure that the simulation behaves similarly to running Langevin dynamics, or if they only need some information about the target density.

We present results for DDS and CMCD using reverse KL (rKL), log-variance divergence (LV), and trajectory balance (TB) on a 40-mixture Gaussian target proposed by Midgley et al. (2023) in Table 4. Our findings reveal the following key observations:

- **Most objectives significantly collapse without Langevin preconditioning.** We note that, at initialization, the samples from the neural samplers already cover all modes, meaning there is no inherent exploration issue. However, even with this favorable initialization, the absence of Langevin preconditioning leads to severe collapse in most objectives.
- **Langevin preconditioning cannot be replaced by alternative target information, such as** $\log p_{\text{target}}$. This suggests that neural samplers require an explicit score term to ensure that the simulation behaves similarly to Langevin dynamics.

Table 5: Sample quality (MMD) by NETS trained with PINN loss (Albergo & Vanden-Eijnden, 2024, Alg 1), both with and without LG in the simulation process during training. As NETS used a different prior and interpolation ($\mathcal{N}(0, 2I)$, mode interpolation) compared to CMCD ($\mathcal{N}(0, 30^2 I)$, geometric interpolation), we present the results by both settings for a fair investigation. N/A suggests diverging.

interpolant	prior	train w. LG	train w/o LG
geom	$\mathcal{N}(0, 2I)$	6.9529	7.0091
	$\mathcal{N}(0, 30^2 I)$	0.3368	0.1721
mode	$\mathcal{N}(0, 2I)$	0.0034	0.0040
	$\mathcal{N}(0, 30^2 I)$	N/A	N/A

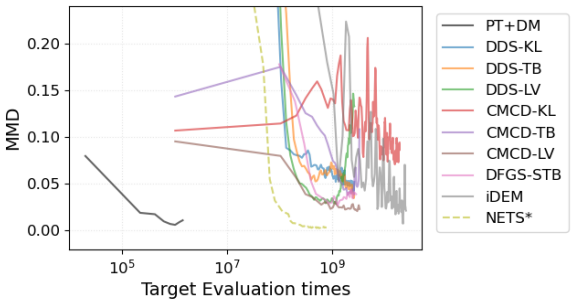


Figure 2: Sample quality vs target evaluation times for different approaches with different objectives on GMM-40 target. *NETS uses mode interpolation, which is distinct from that employed in others.

- **If the initialization is close to optimal, TB and LV refine the solution more stably, while rKL remains prone to mode collapse.** This suggests that future work could explore a training pipeline where the sampler is first warmed up using Langevin dynamics, followed by fine-tuning with these objectives to reduce the number of target energy evaluations during sampling.

We also include results obtained by NETS with the PINN loss in Table 5. Since NETS employs a different prior and interpolation scheme compared to CMCD in Table 4, we present results for both settings to ensure a fair comparison. Surprisingly, we observe that **the PINN loss is relatively robust to Langevin preconditioning during simulation**. Additionally, by design, the PINN loss naturally supports simulation-free training. However, its performance is highly sensitive to the choice of prior and interpolation: a large prior leads to diverging in mode interpolation, while a smaller one also fails under geometric interpolation. Furthermore, the PINN loss requires computing an expensive divergence term, making it challenging to apply to simulation-free approaches with normalizing flows proposed in Section 3.

Finally, the critical role of Langevin preconditioning naturally raises an important question: Is simulation during training with Langevin preconditioning more efficient than directly generating data with Langevin dynamics and fitting a model post hoc? Unfortunately, the answer is no. In Figure 2, we compare several neural samplers against an alternative approach where Parallel Tempering (PT) (PT, Swendsen & Wang, 1986a; Earl & Deem, 2005) is first used to generate samples, followed by fitting a diffusion model. We assess both sample quality and the number of target energy evaluations required. The results clearly show that **almost all neural samplers require several orders of magnitude more target evaluations compared to PT.**

5 Discussions and Conclusions

Motivated by the pursuit of simulation-free training, we reviewed neural samplers from the perspective of sampling processes and training objectives, as well as revisiting their dependence on Langevin preconditioning. Our findings reveal that most training methods for diffusion and control-based neural samplers heavily rely on Langevin preconditioning. While PINN appears to be an exception, it still requires evaluating both the target density and the model's divergence at every time step along the trajectory, making it no more efficient in practice. This highlighted critical caveats in scaling neural samplers to high-dimensional and real-world problems. In fact, while significant advances have been made in learning neural samplers directly from unnormalized densities, the most efficient and practical approach remains running MCMC first and fitting a generative model post hoc.

Our results leave several open questions and reveal some future directions worth exploring. We include a detailed discussion in Appendix A.

REFERENCES

Tara Akhound-Sadegh, Jarrid Rector-Brooks, Joey Bose, Sarthak Mittal, Pablo Lemos, Cheng-Hao Liu, Marcin Sendera, Siamak Ravanbakhsh, Gauthier Gidel, Yoshua Bengio, et al. Iterated denoising energy matching for sampling from boltzmann densities. In *Forty-first International Conference on Machine Learning*, 2024.

Michael S Albergo and Eric Vanden-Eijnden. Nets: A non-equilibrium transport sampler. *arXiv preprint arXiv:2410.02711*, 2024.

Michael S Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*, 2023.

Brian D.O. Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982. ISSN 0304-4149. doi: https://doi.org/10.1016/0304-4149(82)90051-5. URL https://www.sciencedirect.com/science/article/pii/0304414982900515.

Grigory Bartosh, Dmitry Vetrov, and Christian A Naesseth. Neural flow diffusion models: Learnable forward process for improved diffusion modelling. *arXiv preprint arXiv:2404.12940*, 2024.

Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow network based generative models for non-iterative diverse candidate generation. *Advances in Neural Information Processing Systems*, 34:27381–27394, 2021.

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

Denis Blessing, Xiaogang Jia, Johannes Esslinger, Francisco Vargas, and Gerhard Neumann. Beyond elbos: A large-scale evaluation of variational methods for sampling. In *Forty-first International Conference on Machine Learning*, 2024.

George EP Box and George C Tiao. *Bayesian inference in statistical analysis*. John Wiley & Sons, 2011.

Jannis Chemseddine, Christian Wald, Richard Duong, and Gabriele Steidl. Neural sampling from boltzmann densities: Fisher-rao curves in the wasserstein geometry. *arXiv preprint arXiv:2410.03282*, 2024.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Wenlin Chen, Mingtian Zhang, Brooks Paige, José Miguel Hernández-Lobato, and David Barber. Diffusive gibbs sampling, 2024. URL https://arxiv.org/abs/2402.03008.

Julian D Cole. On a quasi-linear parabolic equation occurring in aerodynamics. *Quarterly of applied mathematics*, 9(3):225–236, 1951.

Valentin De Bortoli, Michael Hutchinson, Peter Wirnsberger, and Arnaud Doucet. Target score matching. *arXiv preprint arXiv:2402.08667*, 2024.

Christoph Dellago, Peter G Bolhuis, and Phillip L Geissler. Transition path sampling. *Advances in chemical physics*, 123:1–78, 2002.

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

Arnaud Doucet, Will Sussman Grathwohl, Alexander GDG Matthews, and Heiko Strathmann. Score-based diffusion meets annealed importance sampling. In *Advances in Neural Information Processing Systems*, 2022.

Yuanqi Du, Michael Plainer, Rob Brekelmans, Chenru Duan, Frank Noe, Carla P Gomes, Alan Aspuru-Guzik, and Kirill Neklyudov. Doob's lagrangian: A sample-efficient variational approach to transition path sampling. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

David J Earl and Michael W Deem. Parallel tempering: Theory, applications, and new perspectives. *Physical Chemistry Chemical Physics*, 7(23):3910–3916, 2005.

Wendell H Fleming. Logarithmic transformations with applications in probability and stochastic control. In *Modeling and Control of Systems: in Engineering, Quantum Mechanics, Economics and Biosciences Proceedings of the Bellman Continuum Workshop 1988, June 13–14, Sophia Antipolis, France*, pp. 309–311. Springer, 1989.

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

Louis Grenioux, Maxence Noble, Marylou Gabrié, and Alain Oliviero Durmus. Stochastic localization via iterative posterior sampling. *arXiv preprint arXiv:2402.10758*, 2024.

Carsten Hartmann, Christof Schütte, and Wei Zhang. Jarzynski's equality, fluctuation theorems, and variance reduction: Mathematical analysis and numerical algorithms. *Journal of Statistical Physics*, 175(6):1214–1261, 2019.

Jiajun He, Wenlin Chen, Mingtian Zhang, David Barber, and José Miguel Hernández-Lobato. Training neural samplers with reverse diffusive kl divergence. *arXiv preprint arXiv:2410.12456*, 2024.

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

Eberhard Hopf. The partial differential equation u sub t+ uu sub x= mu sub xx. *Communications on Pure and Applied mathematics*, 3(3):201–230, 1950.

Jian Huang, Yuling Jiao, Lican Kang, Xu Liao, Jin Liu, and Yanyan Liu. Schr {\" o} dinger-f {\" o} llmer sampler: Sampling without ergodicity. *arXiv preprint arXiv:2106.10880*, 2021.

Xunpeng Huang, Hanze Dong, Yifan Hao, Yian Ma, and Tong Zhang. Monte carlo sampling without isoperimetry: A reverse diffusion approach. *arXiv preprint arXiv:2307.02037*, 2023.

Christopher Jarzynski. Nonequilibrium equality for free energy differences. *Physical Review Letters*, 78(14):2690, 1997.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

Salem Lahlou, Tristan Deleu, Pablo Lemos, Dinghuai Zhang, Alexandra Volokhova, Alex Hernández-García, Léna Néhale Ezzine, Yoshua Bengio, and Nikolay Malkin. A theory of continuous generative flow networks, 2023. URL https://arxiv.org/abs/2301.12594.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023.

Weijian Luo, Tianyang Hu, Shifeng Zhang, Jiacheng Sun, Zhenguo Li, and Zhihua Zhang. Diff-instruct: A universal approach for transferring knowledge from pre-trained diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024.

Bálint Máté and François Fleuret. Learning interpolations between boltzmann densities. *arXiv preprint arXiv:2301.07388*, 2023.

Laurence Illing Midgley, Vincent Stimper, Gregor NC Simm, Bernhard Schölkopf, and José Miguel Hernández-Lobato. Flow annealed importance sampling bootstrap. In *The Eleventh International Conference on Learning Representations*, 2023.

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

Kirill Neklyudov, Rob Brekelmans, Daniel Severo, and Alireza Makhzani. Action matching: Learning stochastic dynamics from samples. In *International conference on machine learning*, pp. 25858–25889. PMLR, 2023.

Edward Nelson. *Dynamical Theories of Brownian Motion*. Princeton University Press, 1967. ISBN 9780691079509.

Frank Noé, Simon Olsson, Jonas Köhler, and Hao Wu. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science*, 365(6457):eaaw1147, 2019.

Nikolas Nüsken and Lorenz 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.

RuiKang OuYang, Bo Qiang, and José Miguel Hernández-Lobato. Bnem: A boltzmann sampler based on bootstrapped noised energy matching, 2024. URL https://arxiv.org/abs/2409.09787.

Angus Phillips, Hai-Dang Dau, Michael John Hutchinson, Valentin De Bortoli, George Deligiannidis, and Arnaud Doucet. Particle denoising diffusion sampler. *arXiv preprint arXiv:2402.06320*, 2024.

Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

Teodora Reu, Francisco Vargas, Anna Kerekes, and Michael M Bronstein. To smooth a cloud or to pin it down: Expressiveness guarantees and insights on score matching in denoising diffusion models. In *The 40th Conference on Uncertainty in Artificial Intelligence*, 2024.

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

Lorenz Richter, Ayman Boustati, Nikolas Nüsken, Francisco Ruiz, and Omer Deniz Akyildiz. Vargrad: a low-variance gradient estimator for variational inference. *Advances in Neural Information Processing Systems*, 33:13481–13492, 2020.

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.

Jingtong Sun, Julius Berner, Lorenz Richter, Marius Zeinhofer, Johannes Müller, Kamyar Azizzadenesheli, and Anima Anandkumar. Dynamical measure transport and neural pde solvers for sampling. *arXiv preprint arXiv:2407.07873*, 2024.

Robert H. Swendsen and Jian-Sheng Wang. Replica monte carlo simulation of spin-glasses. *Phys. Rev. Lett.*, 57:2607–2609, Nov 1986a. doi: 10.1103/PhysRevLett.57.2607. URL https://link.aps.org/doi/10.1103/PhysRevLett.57.2607.

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

Saifuddin Syed, Alexandre Bouchard-Côté, George Deligiannidis, and Arnaud Doucet. Nonreversible parallel tempering: a scalable highly parallel mcmc scheme. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 84(2):321–350, 2022.

Nicholas G Tawn, Gareth O Roberts, and Jeffrey S Rosenthal. Weight-preserving simulated tempering. *Statistics and Computing*, 30(1):27–41, 2020.

Yifeng Tian, Nishant Panda, and Yen Ting Lin. Liouville flow importance sampler. In *Forty-first International Conference on Machine Learning*, 2024.

Mark E Tuckerman. *Statistical mechanics: theory and molecular simulation*. Oxford university press, 2023.

Suriyanarayanan Vaikuntanathan and Christopher Jarzynski. Escorted free energy simulations. *The Journal of chemical physics*, 134(5), 2011.

Francisco Vargas, Andrius Ovsianas, David Fernandes, Mark Girolami, Neil D Lawrence, and Nikolas Nüsken. Bayesian learning via neural schr\" odinger-f\" ollmer flows. *arXiv preprint arXiv:2111.10510*, 2021.

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

Francisco Vargas, Shreyas Padhy, Denis Blessing, and Nikolas Nusken. Transport meets variational inference: Controlled monte carlo diffusions. In *The Twelfth International Conference on Learning Representations*, 2024.

Dawn Woodard, Scott Schmidler, and Mark Huber. Sufficient Conditions for Torpid Mixing of Parallel and Simulated Tempering. *Electronic Journal of Probability*, 14(none):780 – 804, 2009. doi: 10.1214/EJP.v14-638. URL https://doi.org/10.1214/EJP.v14-638.

Dinghuai Zhang, Ricky TQ Chen, Cheng-Hao Liu, Aaron Courville, and Yoshua Bengio. Diffusion generative flow samplers: Improving learning signals through partial trajectory optimization. *The Twelfth 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.

Wei Zhang. Some new results on relative entropy production, time reversal, and optimal control of time-inhomogeneous diffusion processes. *Journal of Mathematical Physics*, 62(4), 2021.

A Further Discussion

Our results reveal several open questions and future directions worth exploring:

First, talking about neural samplers, many works focus on learning models directly from the unnormalized density, avoiding the use of any data from the target density. However, given that the Langevin preconditioning plays a crucial role in most approaches, we may equivalently interpret the training process as running several steps of MCMC to obtain approximate samples. This interpretation, blurring the distinction between data-driven and data-free approaches, challenges the definition of these "data-free" neural samplers. Furthermore, as our results demonstrate, a straightforward two-step approach—first running Parallel Tempering (PT) to obtain samples, followed by fitting a diffusion model—yields significantly higher efficiency compared to nearly all neural samplers. This observation further questions the practical justification and motivation of "data-free" neural samplers. Therefore, rather than attempting to completely avoid the use of data, **a more promising and practical direction may involve developing objective functions or training pipelines that rely on a limited amount of data for a more efficient acquisition of information from each target density evaluation.**

However, we emphasize that while we advocate for the explicit utilization of data, we acknowledge that **it may not always be feasible, or even reasonable, for newly developed approaches to surpass these well-established baselines from the outset.** The methods developed within "data-free" training pipelines remain valuable and can provide inspiration for approaches that more effectively leverage data, potentially leading to improved efficiency and performance in neural samplers.

Based on our observations, PINN loss appears to be an example with such potential. It demonstrates greater robustness in the absence of Langevin preconditioning and naturally supports simulation-free training by its design. However, it still requires extensive target evaluations along the entire trajectory and tends to be more sensitive to hyperparameters. Therefore, **future research could focus on learning better priors or interpolations.** A straightforward approach may involve first obtaining approximate samples from the target distribution using methods such as MCMC, then learning priors or interpolants from these samples, and finally leveraging the learned hyperparameters to refine the sample quality, in an iterative manner.

B Taxonomy of Objective Functions

In this section, we briefly describe different objectives that we reviewed and used in the main text.

B.1 Path measure alignment objectives

The *path measure alignment* framework aims to align the sampling process starting from p_{prior} to a "target" process starting from p_{target}. In the following, we denote $\mathbb{Q}$ as the sampling process and $\mathbb{P}$ as the "target" process. However, we should note that this notation does not necessarily imply that $\mathbb{Q}$ is the process parameterized by the model. In fact, this is only true for samplers like PIS or DDS. For escorted transport samplers like CMCD, both $\mathbb{Q}$ and $\mathbb{P}$ involve the model, and for annealed variance reduction sampler like MCD, $\mathbb{Q}$ is fixed, and the model only appears in $\mathbb{P}$. We now describe five commonly used objectives:

Reverse KL divergence. Reverse KL divergence is defined as

$$D_{\mathrm{KL}}[\mathbb{Q}||\mathbb{P}] = \mathbb{E}_{\mathbb{Q}}\left[\log \frac{\mathrm{d}\mathbb{Q}}{\mathrm{d}\mathbb{P}}\right]. \tag{15}$$

In practice, we approximate the expectation with Monte Carlo estimators, and calculate the log Radon–Nikodym derivative $\log \frac{\mathrm{d}\mathbb{Q}}{\mathrm{d}\mathbb{P}}$, either through Gaussian approximation via Euler–Maruyama discretization or by applying Girsanov's theorem.

Log-variance divergence. Log-variance divergence optimizes the second moment of the log ratio

$$D_{\mathrm{logvar}}[\mathbb{Q}||\mathbb{P}] = \mathrm{Var}_{\tilde{\mathbb{Q}}}\left(\log \frac{\mathrm{d}\mathbb{Q}}{\mathrm{d}\mathbb{P}}\right). \tag{16}$$

Unlike KL divergence, which requires the expectation to be taken with respect to $\mathbb{Q}$, log-variance allows the variance to be computed under a different measure $\tilde{\mathbb{Q}}$. This flexibility suggests that we can detach the gradient of the trajectory or utilize a buffer to stabilize training. On the other hand, when the variance is taken under $\mathbb{Q}$, the gradient of log-variance divergence w.r.t parameters in $\mathbb{Q}$ is the same as that of reverse KL divergence (Richter et al., 2020):

$$\frac{\mathrm{d}}{\mathrm{d}\theta}\mathrm{Var}_{\tilde{\mathbb{Q}}}\left(\log\frac{\mathrm{d}\mathbb{Q}_\theta}{\mathrm{d}\mathbb{P}}\right)\Bigg|_{\tilde{\mathbb{Q}}=\mathbb{Q}_\theta} = \frac{\mathrm{d}}{\mathrm{d}\theta}D_{\mathrm{KL}}[\mathbb{Q}_\theta||\mathbb{P}]. \tag{17}$$

However, we note that this conclusion holds only *in expectation*. In practice, when the objective is calculated with Monte Carlo estimators, they will exhibit different behavior.

Trajectory balance. Trajectory balance optimizes the squared log ratio

$$D_{\mathrm{TB}}[\mathbb{Q}||\mathbb{P}] = \mathbb{E}_{\tilde{\mathbb{Q}}}\left[\left(\log\frac{\mathrm{d}\mathbb{Q}}{\mathrm{d}\mathbb{P}} - k\right)^2\right], \tag{18}$$

which is equivalent to the log-variance divergence with a learned baseline k.

Sub-trajectory balance. TB loss matches the entire $\mathbb{Q}$ and $\mathbb{P}$ as a whole. Alternatively, we can match segments of each trajectory individually to ensure consistency across the entire trajectory. This approach leads to the sub-trajectory balance objective. For simplicity, though it is possible to define sub-trajectory balance in continuous time, we define it with time discretization.

With Euler–Maruyama discretization, we discretize $\mathbb{Q}$ and $\mathbb{P}$ into sequential produce of measure, with density given by:

$$p_0(X_0)\prod_{n=0}^{N-1}p_F(X_{n+1}|X_n) \quad \text{and} \quad \tilde{p}_{\text{target}}(X_N)\prod_{t=0}^{N-1}p_B(X_n|X_{n+1}). \tag{19}$$

Note that the density for discretized $\mathbb{P}$ can be unnormalized.

Then, we introduce a sequence of intermediate densities $\{\pi_n\}_{n=0}^N$, where the boundary conditions are given by $\pi_0 = p_{\text{prior}}$ and $\pi_N = \tilde{p}_{\text{target}}$. These intermediate distributions can either be prescribed as a fixed interpolation between the target and prior distributions or be learned adaptively through a parameterized neural network.

Finally, we define the sub-trajectory balance objective as

$$D_{\mathrm{STB}}[\mathbb{Q}||\mathbb{P}] = \mathbb{E}_{\tilde{\mathbb{Q}}}\left[\sum_{0\le i<j\le N}\left(\log\frac{\pi_i(x_i)\prod_{n=i}^{j-1}p_F(x_{n+1}|x_n)}{\pi_j(x_j)\prod_{n'=i}^{j-1}p_B(x_{n'}|x_{n'+1})} + k_i - k_j\right)^2\right]. \tag{20}$$

Detailed balance. Detailed balance can be viewed as an extreme case of sub-trajectory balance, where instead of summing over sub-trajectories of all lengths, we only calculate the sub-trajectory balance over each discretization step:

$$D_{\mathrm{DB}}[\mathbb{Q}||\mathbb{P}] = \mathbb{E}_{\tilde{\mathbb{Q}}}\left[\sum_{0\le i\le N-1}\left(\log\frac{\pi_i(x_i)p_F(x_{i+1}|x_i)}{\pi_j(x_{i+1})p_B(x_i|x_{i+1})} + k_i - k_{i+1}\right)^2\right]. \tag{21}$$

B.2 Marginal alignment objectives

Unlike path measure alignment, *marginal alignment* objectives directly enforce the sampling process at each time step t to match with some marginal π_t. π_t can be either prescribed as an interpolation between the target and prior, with boundary conditions $\pi_0 = p_{\text{prior}}$ and $\pi_T = p_{\text{target}}$, or be learned through a network under the constraint of the boundary conditions. Commonly used objectives in this framework include PINN and action matching:

PINN. For the sampling process defined by $dX_t = \left(f_\theta(X_t, t) + \sigma_t^2 \nabla \log \pi_t(X_t)\right) dt + \sigma_t \sqrt{2} dW_t$, the PINN loss is given by

$$\mathcal{L}_{\text{PINN}} = \int_0^T \mathbb{E}_{\tilde{q}_t(X_t)} ||\nabla \cdot f_\theta(X_t, t) + \log \pi_t(X_t) \cdot f_\theta(X_t, t) + (\partial_t \log \pi_t)(X_t) + \partial_t F(t)||^2 dt, \tag{22}$$

where $F(t)$ is parameterized by a neural network. Note that the expectation can be taken over an arbitrary $\tilde{q}_t$, as long as the marginal of $\mathbb{Q}$ at time t is absolute continuous to $\tilde{q}_t$. We also note PINN does not depend on the specific value of σ_t in the sampling process.

Action matching. Similar to PINN, an action matching-based (Neklyudov et al., 2023) objective is derived by (Albergo & Vanden-Eijnden, 2024) for the PDE-constrained optimization problem

$$\mathcal{L}_{\text{AM}} = \int_0^T \mathbb{E}_{q_t(X_t)} \left[\frac{1}{2} ||\nabla \phi_t(X_t)||^2 + \partial_t \phi_t(X_t)\right] dt + \mathbb{E}_{p_{\text{prior}}(X_0)} \left[\phi_0(X_0)\right] - \mathbb{E}_{p_{\text{target}}(X_T)} \left[\phi_T(X_T)\right], \tag{23}$$

where the vector field $b_t = \nabla \phi_t$, induced by a scalar potential, and ϕ_t is called the "action".

C Detailed Summary of Samplers

In this section, we provide a more detailed review of diffusion and control-based neural samplers. We also discuss how these neural samplers rely on the Langevin preconditioning in the end.

C.1 Sampling process and Objectives

We write the sampling process as follows:

$$dX_t = \left[\mu_t(X_t) + \sigma_t^2 b_t(X_t)\right] dt + \sigma_t \sqrt{2} dW_t, \quad X_0 \sim p_{\text{prior}}, \tag{24}$$

(1) Path Integral Sampler (PIS, Zhang & Chen, 2022) and concurrently (NSFS, Vargas et al., 2021): PIS fixes $p_{\text{prior}} = \delta_0, \sigma_t = 1/\sqrt{2}$ and learns a network $f_\theta(\cdot) = \mu_t(\cdot) + \sigma_t^2 b_t(\cdot)$ so that Equation (24) approximate the time-reversal of the following SDE (Pinned Brownian Motion):

$$dY_t = -\frac{Y_t}{T-t} dt + dW_t, \quad Y_0 \sim p_{\text{target}}. \tag{25}$$

We define Equation (25) as the time-reversal of Equation (24) when $Y_t \sim X_{T-t}$. The network is learned by matching the reverse KL (Zhang & Chen, 2022; Vargas et al., 2021) or log-variance divergence (Richter & Berner, 2024) between the path measures of the sampling and the target process.

(2) Diffusion generative flow samplers (DFGS, Zhang et al., 2024) learns to sample from the same process as PIS, but with a new introduction of local objectives including detailed balance and (sub-)trajectory balance. In fact, trajectory balance can been shown to be equivalent to the log-variance objective with a learned baseline rather than a Monte Carlo (MC) estimator for the first moment (Nüsken & Richter, 2021).

(3) Denoising Diffusion Sampler (DDS, Vargas et al., 2023) and time-reversed Diffusion Sampler (DIS, Berner et al., 2024): both DDS and DIS fix $\mu_t(X_t, t) = \beta_{T-t} X_t, \sigma_t = v\sqrt{\beta_{T-t}}, p_{\text{prior}} = \mathcal{N}(0, v^2 I)$, and learn a network $f_\theta(\cdot, t) = b_t(\cdot, t)/2$ so that Equation (24) approximates the time-reversal of the VP-SDE:

$$dY_t = -\beta_t Y_t dt + v\sqrt{2\beta_t} dW_t, \quad Y_0 \sim p_{\text{target}}. \tag{26}$$

Similar to PIS, the network can be trained either with reverse KL divergence or log-variance divergence. In an optimal solution, f_θ will approximate the score $f_\theta(\cdot, t) \approx \nabla \log p_{T-t}(\cdot)$, where $p_t(X) = \int \mathcal{N}(X | \sqrt{1-\lambda_t} Y, v^2 \lambda_t I) p_{\text{target}}(Y) dY$ and $\lambda_t = 1 - \exp(-2\int_0^t \beta_s ds)$.

(4) Iterated Denoising Energy Matching (iDEM, Akhound-Sadegh et al., 2024): iDEM fixes $\mu_t(X_t, t) = 0, p_{\text{prior}} = \mathcal{N}(0, T^2 I)$, and learns a network $f_\theta(\cdot, t) = b_t(\cdot, t)/2$ to approximate

the score $f_\theta(X_t, t) \approx \nabla \log p_{T-t}(X_t)$. This is achieved by writing the score with target score identity (TSI, De Bortoli et al., 2024), and estimating it with a self-normalized importance sampler:

$$\nabla \log p_{T-t}(X_t) \overset{\text{TSI}}{=} \int p_{T|T}(X_T|X_t) \nabla \log \tilde{p}_{\text{target}}(X_T) dX_T \tag{27}$$

$$\overset{\text{Bayes' Rule}}{=} \int \frac{\tilde{p}_{\text{target}}(X_T) p_{t|T}(X_t|X_T)}{\int \tilde{p}_{\text{target}}(X_T) p_{t|T}(X_t|X_T) dX_T} \nabla \log \tilde{p}_{\text{target}}(X_T) dX_T \tag{28}$$

$$= \int q_{T|t}(X_T|X_t) \frac{\tilde{p}_{\text{target}}(X_T) p_{t|T}(X_t|X_T) \nabla \log \tilde{p}_{\text{target}}(X_T)}{q_{T|t}(X_T|X_t) \int q_{T|t}(X_T|X_t) \frac{\tilde{p}_{\text{target}}(X_T) p_{t|T}(X_t|X_T)}{q_{T|t}(X_T|X_t)} dX_T} dX_T. \tag{29}$$

By choosing $q_{T|t}(X_T|X_t) \propto p_{t|T}(X_t|X_T)$, we obtain

$$\nabla \log p_{T-t}(X_t) = \int q(X_T|X_t) \frac{\tilde{p}_{\text{target}}(X_T) \nabla \log \tilde{p}_{\text{target}}(X_T)}{\int q(X_T|X_t) \tilde{p}_{\text{target}}(X_T) dX_T} dX_T \tag{30}$$

$$\approx \sum_n \frac{\tilde{p}_{\text{target}}(X_T^{(n)})}{\sum_n \tilde{p}_{\text{target}}(X_T^{(n)})} \nabla \log \tilde{p}_{\text{target}}(X_T^{(n)}), \quad X_T^{(n)} \sim q_{T|t}(X_T|X_t) \tag{31}$$

$$=: \nabla \log \widehat{p_{T-t}}(X_t). \tag{32}$$

Then, iDEM matches $f_\theta(X_t, t)$ with $\nabla \log \widehat{p_{T-t}}(X_t)$ by L^2 loss. In optimal, the sampling process will approximate the time-reversal of a VE-SDE:

$$dY_t = \sqrt{2t} dW_t, \quad Y_0 \sim p_{\text{target}}. \tag{33}$$

Several extensions have been developed based on iDEM: Bootstrapped Noised Energy Matching (BNEM, OuYang et al., 2024) generalizes the self-normalized importance sampling estimator of the score to the energy function, enabling the training of energy-parameterized diffusion models. They also proposed a bootstrapping approach to reduce the training variance. Diffusive KL (DiKL, He et al., 2024) integrates this estimator with variational score distillation techniques (Poole et al., 2022; Luo et al., 2024) to train a one-step generator as the neural sampler. Also, DiKL proposes using MCMC to draw samples from $p_{T|t}(X_T|X_t)$ to estimate the score, instead of relying on the self-normalized importance sampling estimator with the proposal $q_{T|t}(X_T|X_t) \propto p_{t|T}(X_t|X_T)$, leading to lower variance during training.

We also note that iDEM's score estimator is closely related to stochastic control problems. One can re-express the estimator regressed in iDEM in terms of the optimal drift of a stochastic control problem (Huang et al., 2021), the optimal control f^* can be expressed in terms of the score (e.g. See Remark 3.5 in Reu et al. (2024)) :

$$f_t^*(X_t) = -\nabla \log \phi_{T-t}(X_t) = -\nabla \ln \nu_{T-t}^{\text{ref}}(X_T) + \nabla \log p_{T-t}(X_t), \tag{34}$$

where $\phi_t(X_t)$ is the value function, which can be expressed as a conditional expectation via the Feynman-Kac formula followed by the Hopf-Cole transform (Hopf, 1950; Cole, 1951; Fleming, 1989):

$$\phi_t(x) = \mathbb{E}_{X_T \sim q_{T|t}(X_T|x)} \left[\frac{\tilde{p}_{\text{target}}}{\nu_T^{\text{ref}}}(X_T) \right]. \tag{35}$$

Where in the case for VE-SDE (i.e. iDEM) and $\nu_t^{\text{ref}}(x) = \mathcal{N}(x|0, t + \sigma_{\text{prior}}^2)$ and thus $\phi_{T-t}(X_t) = \frac{X_t}{T-t+\sigma_{\text{init}}^2} + \nabla \log p_{T-t}(X_t)$.

Note the MC Estimator of $\nabla \log \phi_{T-t}(X_t)$ (e.g. Equation 8) was used in Schrödinger-Föllmer Sampler (SFS, Huang et al., 2021) to sample from time-reversal of pinned Brownian Motion, yielding an akin estimator to the one used in iDEM, in particular they carry out an MC estimator of the following quantity:

$$\nabla \phi_{T-t}(x) = \frac{\mathbb{E}_{Z \sim \mathcal{N}(0,I)} \left[\nabla \frac{\tilde{p}_{\text{target}}}{\nu_T^{\text{ref}}}(\mu_{T|T-t} x + \sigma_{T|T-t} Z) \right]}{\mathbb{E}_{Z \sim \mathcal{N}(0,I)} \left[\frac{\tilde{p}_{\text{target}}}{\nu_T^{\text{ref}}}(\mu_{T|T-t} x + \sigma_{T|T-t} Z) \right]}. \tag{36}$$

Where, we have assumed that $q_{T|t}(x_T|x_t) = \mathcal{N}(x_T|\mu_{T|t}x_t, \sigma_{T|t})$ as is the case with most time reversal based samplers and generative models.

(5) Monte Carlo Diffusion (MCD, Doucet et al., 2022): unlike other neural samplers, MCD's sampling process is fixed as $\mu_t = 0, \sigma_t = 1, b_t(X_t, t) = \nabla \log \pi_t(X_t)$, where π_t is the geometric interpolation between target and prior, i.e., $\pi_t(X_t) = p^{\beta_t}_{\text{target}}(X_t)p^{1-\beta_t}_{\text{prior}}(X_t)$. It can be viewed as sampling with AIS using ULA as the kernel. Note, that this transport is non-equilibrium, as the density of X_t is not necessary $\pi_t(X_t)$. Therefore, MCD trains a network to approximate the time-reversal of the forward process and perform importance sampling (more precisely, AIS) to correct the bias of the non-equilibrium forward process.

(6) Controlled Monte Carlo Diffusion (CMCD, Vargas et al., 2024) and Non-Equilibrium Transport Sampler (NETS, Máté & Fleuret, 2023; Albergo & Vanden-Eijnden, 2024): Similar to MCD, CMCD and NETS also set $b_t(X_t, t) = \nabla \log \pi_t(X_t)$ and π_t is the interpolation between target and prior. Different from MCD where the sampling process is fixed, CMCD and NETS learn $f_\theta(\cdot, t) = \mu_t(\cdot, t)$ so that the marginal density of samples X_t simulated by Equation (2) will approximate π_t. As a special case, Liouville Flow Importance Sampler (LFIS, Tian et al., 2024) fixes $\sigma_t = 0$ and learns an ODE to transport between π_t.

C.2 Lanvegin Preconditioning in Diffusion/Control-based Neural Samplers

Explicit Langevin preconditioning. In samplers including PIS, DDS, DFGS, DIS, etc., The network is parameterized with a skip connection using Lanvegin preconditioning:

$$f_\theta(\cdot, t) = \text{NN}_{1,\theta}(\cdot, t) + \text{NN}_{2,\theta}(t) \circ \nabla \log p_{\text{target}}(\cdot). \tag{37}$$

In samplers like MCD, the forward process is a sequence of Lanvegin dynamics with invariant density π_t as the interpolation between prior and target. In CMCD and NETS, the drift of forward process is given by the network output plus a score term:

$$f_\theta(X_t, t) + \sigma_t^2 \nabla \log \pi_t(X_t). \tag{38}$$

Implicit Langevin preconditioning. In iDEM, we regress the network with

$$\nabla \log p_{T-t}(X_t) \approx \sum_n \frac{\tilde{p}_{\text{target}}(X_T^{(n)})}{\sum_n \tilde{p}_{\text{target}}(X_T^{(n)})} \nabla \log \tilde{p}_{\text{target}}(X_T^{(n)}), \quad X_T^{(n)} \sim q_{T|t}(X_T|X_t). \tag{39}$$

Note that while the network does not explicitly depend on the score of the target density, the objective compels it to learn gradient information. This gradient information is utilized during simulation when collecting the buffer every few iterations, effectively inducing an implicit Langevin preconditioning.

No Langevin preconditioning. LFIS does not rely on Langevin preconditioning during simulation. Like NETS, it employs the PINN loss, but its sampling process is governed by an ODE. Thus, similar to our discussion in the main text on eliminating Langevin preconditioning for PINN-based CMCD, LFIS inherently removes this dependency in its design.

LFIS adopts several tricks to stabilize the training and ensure mode covering: it learns the ODE drift progressively, starting from the prior and gradually transitioning to the target. Additionally, it employs separate networks for different time steps to prevent forgetting. But even without these tricks, our results in Table 5 confirm the robustness of the PINN loss to Langevin preconditioning when the interpolation and prior are carefully tuned.

D NF-DDS

Here we present a derivation of the NF-DDS objective.

$$
\begin{aligned}
& D_{\mathrm{KL}}[\mathbb{Q}||\mathbb{P}] \\
=& \mathbb{E}_{\mathbb{Q}}\left[\int_0^T \frac{1}{4v^2\beta_t}\|\tilde{F}_\theta(Y_t, T-t) - v^2\beta_t\nabla\log q_\theta(Y_t, T-t) - \beta_t Y_t\|^2 dt\right] + D_{\mathrm{KL}}[q_\theta(\cdot, T)||p_{\text{target}}] \\
=& \int_0^T \mathbb{E}_{\mathbb{Q}}\left[\frac{1}{4v^2\beta_t}\|\tilde{F}_\theta(Y_t, T-t) - v^2\beta_t\nabla\log q_\theta(Y_t, T-t) - \beta_t Y_t\|^2\right] dt + D_{\mathrm{KL}}[q_\theta(\cdot, T)||p_{\text{target}}] \\
=& \int_0^T \frac{1}{4v^2\beta_t}\mathbb{E}_{q_\theta(Y,T-t)}\|\tilde{F}_\theta(Y, T-t) - v^2\beta_t\nabla\log q_\theta(Y, T-t) - \beta_t Y\|^2 dt + D_{\mathrm{KL}}[q_\theta(\cdot, T)||p_{\text{target}}] \\
=& \int_0^T \frac{1}{4v^2\beta_{T-t}}\mathbb{E}_{q_\theta(Y,t)}\|\tilde{F}_\theta(Y, t) - v^2\beta_{T-t}\nabla\log q_\theta(Y, t) - \beta_{T-t} Y\|^2 dt + D_{\mathrm{KL}}[q_\theta(\cdot, T)||p_{\text{target}}].
\end{aligned} \tag{40}
$$

E NF-CMCD

In this section, we proposed a CMCD variation with normalizing flow for simulation-free training.

In CMCD, we match the forward sampling process:

$$
dX_t = \left(\tilde{F}_\theta(X_t, t) + \sigma_t^2\nabla\log q_\theta(X_t, t)\right)dt + \sigma_t\sqrt{2}dW_t, X_0 \sim q_\theta(X_0, 0), \tag{41}
$$

with a target backward process, calculated by Nelson's condition assuming the marginal of the SDE at each time step matches with a prescribed marginal density e.g. $\pi_t(\cdot) = p_{\text{target}}^{\beta_t}(\cdot)p_{\text{prior}}^{1-\beta_t}(\cdot)$:

$$
\begin{aligned}
dY_t = -\Big(&\tilde{F}_\theta(Y_t, T-t) + \sigma_{T-t}^2\nabla\log q_\theta(Y_t, T-t) \\
&- 2\sigma_{T-t}^2\nabla\log\pi_{T-t}(Y_t, T-t)\Big)dt + \sigma_{T-t}\sqrt{2}dW_t, Y_0 \sim p_{\text{target}}.
\end{aligned} \tag{42}
$$

Again, similar to NF-DDS, the time-reversal of Equation (41) can be calculated by Nelson's condition:

$$
dY_t = -\left(\tilde{F}_\theta(Y_t, T-t) - \sigma_{T-t}^2\nabla\log q_\theta(Y_t, T-t)\right)dt + \sigma_{T-t}\sqrt{2}dW_t, Y_0 \sim q_\theta(Y_0, T). \tag{43}
$$

By Girsanov theorem, the KL divergence between the path measure by Equation (43) (denoted as $\mathbb{Q}$) and Equation (42) (as $\mathbb{P}$) is:

$$
D_{\mathrm{KL}}[\mathbb{Q}||\mathbb{P}] = \int_0^T \frac{1}{\sigma_t}\mathbb{E}_{q_\theta(Y,t)}\|\sigma_t^2\nabla\log q_\theta(Y, t) - \sigma_t^2\nabla\log\pi_t(Y, t)\|^2 dt + D_{\mathrm{KL}}[q_\theta(\cdot, T)||p_{\text{target}}]. \tag{44}
$$

This coincides with the Fisher divergence between each marginal.

After training, we can sample from

$$
dX_t = \left(\tilde{F}_\theta(X_t, t) + \sigma_t^2\nabla\log\pi_t(X_t)\right)dt + \sigma_t\sqrt{2}dW_t, X_0 \sim q_\theta(X_0, 0), \tag{45}
$$

with approximate reversal

$$
dY_t = -\left(\tilde{F}_\theta(Y_t, T-t) - \sigma_{T-t}^2\nabla\log\pi_{T-1}(Y_t)\right)dt + \sigma_{T-t}\sqrt{2}dW_t, Y_0 \sim \tilde{p}_{\text{target}}. \tag{46}
$$

F Additional Experiment Details

F.1 Evaluation Metrics

In this paper, we evaluate the samples quality by ELBO, EUBO and MMD. The ELBO (Evidence Lower Bound) is a lower bound of the (log) normalization factor, reflecting how well the model is

concentrated within each mode; on the other hand, EUBO (Evidence Upper Bound, Blessing et al., 2024) provides an upper bound, representing if the model successfully covers all modes.

The MMD (Maximum Mean Discrepancy) measures the distributional discrepancy between the generated samples and the target distribution. We base our MMD implementation on the code by Chen et al. (2024) at `https://github.com/Wenlin-Chen/DiGS/blob/master/mmd.py`, using 10 kernels and fixing the `sigma = 100`.

For all experiments, we evaluate ELBO, EUBO and MMD with 10000 samples.

F.2 Hyperparameters

Table 6: Hyperparameters used for experiments.

method	objective	prior	lr	precond	network size
DDS	rKL	$\mathcal{N}(0, 30^2 I)$	5e-4	LG	[64, 64]
	LV	$\mathcal{N}(0, 30^2 I)$	5e-4	LG	[64, 64]
	TB	$\mathcal{N}(0, 30^2 I)$	5e-4	LG	[64, 64]
	rKL	$\mathcal{N}(0, 30^2 I)$	5e-4	- / $\log p_{\text{target}}$	[256, 256, 256, 256, 256]
	LV	$\mathcal{N}(0, 30^2 I)$	5e-4	- / $\log p_{\text{target}}$	[256, 256, 256, 256, 256]
	TB	$\mathcal{N}(0, 30^2 I)$	5e-4	- / $\log p_{\text{target}}$	[256, 256, 256, 256, 256]
CMCD	rKL	$\mathcal{N}(0, 30^2 I)$	5e-4	LG	[64, 64]
	LV	$\mathcal{N}(0, 30^2 I)$	5e-3	LG	[64, 64]
	TB	$\mathcal{N}(0, 30^2 I)$	5e-4	LG	[64, 64]
	rKL	$\mathcal{N}(0, 30^2 I)$	5e-4	-	[256, 256, 256, 256, 256]

We summarize the hyperparameters for DDS and CMCD in Table 6. These hyperparameters are chosen according to Blessing et al. (2024). For PINN-based experiments shown in Table 5, we follow the hyperparameter used in NETS (Albergo & Vanden-Eijnden, 2024), including network size, learning rate and its schedule. etc.

G Additional Experimental Results

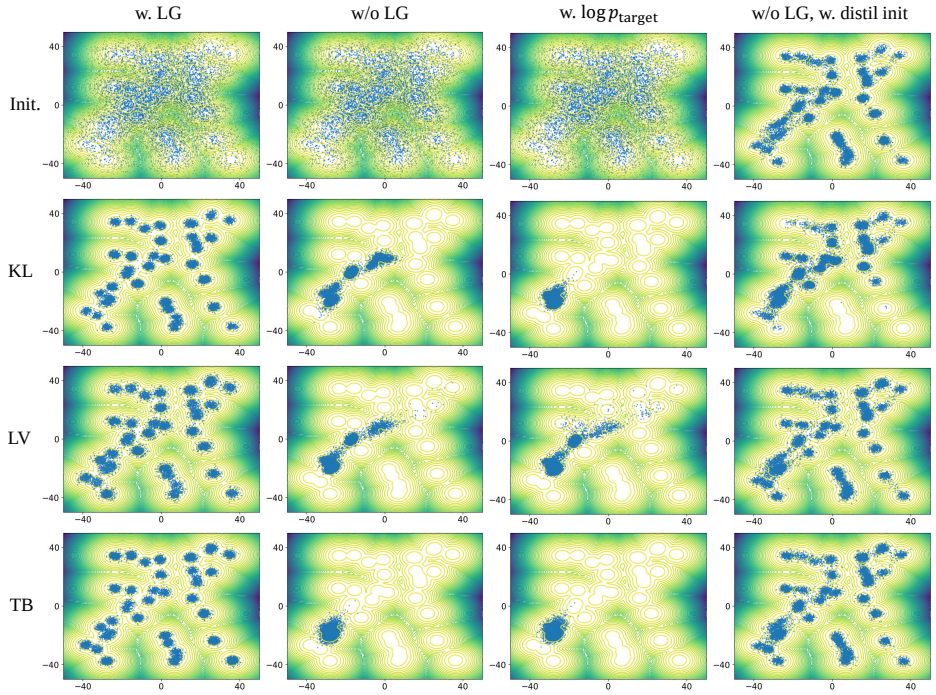


Figure 3: Sampled obtained by DDS with different settings. The first line shows the initialization.

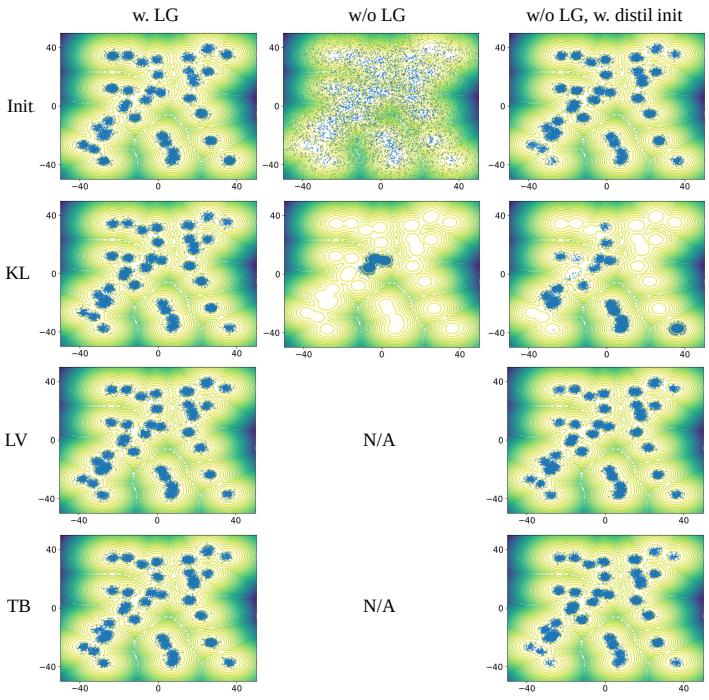


Figure 4: Sampled obtained by CMCD with different settings. The first line shows the initialization and N/A indicates diverging. We can see when trained with Langevin preconditioning, we can see that CMCD already captures modes after initialization.

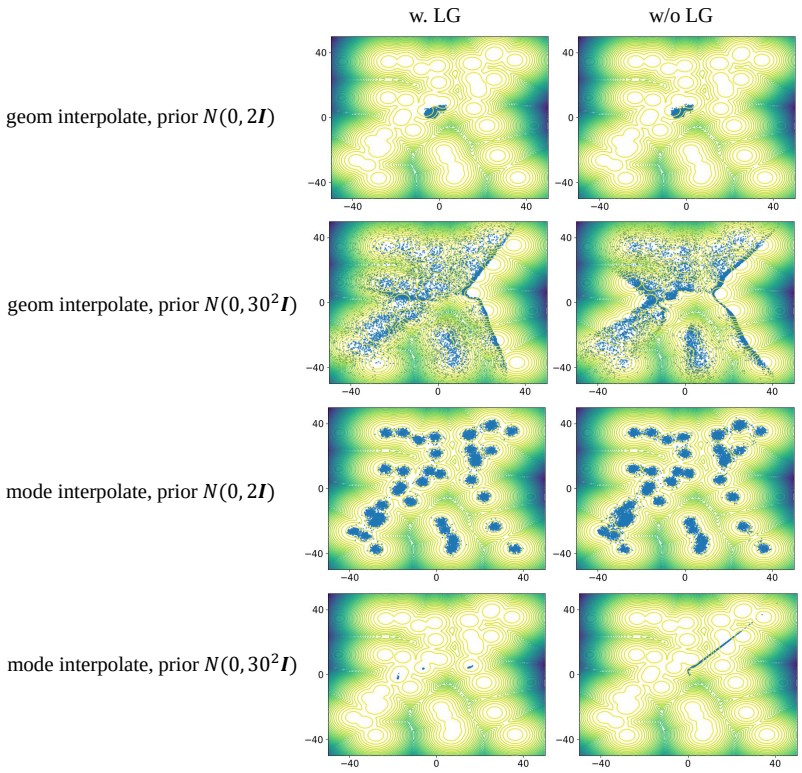


Figure 5: Sampled obtained by PINN with different settings. As we can see, PINN seems to be highly robust to Langevin preconditioning. However, it is more sensitive to the choice of prior and interpolation.