Workshop at the 7th Symposium on Advances in Approximate Bayesian Inference (non-archival), 2025 1–16

Diffusion-based Sampling via Amortized Posterior Inference

Yi Han YH3660@COLUMBIA.EDU
Luhuan Wu LW2827@COLUMBIA.EDU
John P. Cunningham JPC2181@COLUMBIA.EDU
Department of Statistics, Columbia University

Abstract

Recent years have seen a surge of interest in utilizing diffusion models for sampling from an unnormalized target density without access to data. A common approach is to construct a reverse diffusion SDE whose terminal distribution matches the desired target. In this work, we propose a novel method that trains a diffusion model to sample from unnormalized distributions by reformulating the sampling problem as a sequence of posterior inference tasks. To effectively solve these inference problems, we leverage existing Monte Carlo methods and design a training objective based on the inference results. Empirically, our method demonstrates competitive performance on Gaussian mixture densities and achieves better performance on the DW-4 particle system, compared to other diffusion-based sampling methods. These results highlight the potential of integrating diffusion models with advanced posterior inference techniques for improved sampling in complex distributions.

1. Introduction

Sampling from an unnormalized target distribution is a fundamental problem in many applications, ranging from Bayesian inference to simulating molecular systems. Simulation-based methods such as Markov chain Monte Carlo (Brooks, 1998), importance sampling (Tokdar and Kass, 2010), and sequential Monte Carlo (Doucet et al., 2001) are classic ways to draw approximate samples in a large compute limit. However, they can suffer from slow convergence and poor scalability to high-dimensional problems.

Diffusion models have recently emerged as a promising tool for sampling problems. Standard diffusion models learn the target distribution via a neural network-parameterized stochastic differential equation (SDE) trained on a large dataset (Ho et al., 2020; Song et al., 2020). For sampling tasks where such data is unavailable, recent works form various approximations to the SDE directly using the unnormalized target (Huang et al., 2023; He et al., 2024b; Grenioux et al., 2024). While flexible, they require expensive Monte Carlo (MC) simulations at inference time and introduce approximation biases. Other approaches train diffusion models to enable faster sampling (Akhound-Sadegh et al., 2024), though they may encounter challenges with high-variance training objectives (Noble et al., 2024).

In this work, we introduce a novel method for training diffusion models to sample from unnormalized target distributions by amortizing posterior inference. Building on prior works, we first identify the sampling problem as a sequence of posterior inference tasks, each indexed by a varying noise level. At each noise level, we define a posterior distribution by conditioning the target (acting as a prior) on a Gaussian noising likelihood introduced by the forward process of diffusion models. While these posteriors remain intractable, due to Gaussian noise conditioning, they are easier to approach with existing inference techniques

compared to the original sampling problem. We then train an amortized neural network targeting the posterior mean. This network effectively parametrizes a diffusion model, enabling efficient sampling from the unnormalized target distribution.

We evaluate the empirical performance of our method on Gaussian mixtures and double-well potentials associated with n-body particle systems (DW4, Köhler et al., 2020). Our method achieves competitive performance with other diffusion-based sampling methods on Gaussian mixture densities while outperforming others on most metrics for DW-4.

To summarize, our contributions are:

1. A novel method that trains a diffusion model to sample from unnormalized distributions by amortizing a sequence of posterior inference tasks.
2. Empirical evaluation of Gaussian mixture densities and the DW-4 particle system, showing competitive or superior performance to other diffusion-based methods.

2. Background

Diffusion models (Ho et al., 2020; Song et al., 2020) define a forward SDE that transforms a target distribution $p(x_0)$ at $t = 0$ into a reference distribution $p(x_1)$ at $t = 1$:

$$dx_t = f(t)x_t\,dt + g(t)\,dB_t, \tag{1}$$

where B_t is the standard Brownian motion, $f(t)x_t$ is the drift coefficient, and $g(t)$ is the diffusion coefficient. See common diffusion parameterizations in Appendix A.

To generate a sample from $p(x_0)$, one simulates from the reverse process which starts with $x_1 \sim p(x_1)$ and evolves the sample through the reverse-time SDE:

$$dx_t = \left[f(t)x_t \;-\; g(t)^2\,\nabla_{x_t} \log p(x_t)\right]dt \;+\; g(t)\,dB_t, \tag{2}$$

where $\nabla_{x_t} \log p(x_t)$ is the gradient of the marginal density $p(x_t)$ at time t, known as the score function. Typically, the score function is intractable and hence is approximated by a score network trained using samples from $p(x_0)$.

3. Method

Our goal is to sample from a target distribution $p(x_0)$, whose density is known up to a normalization constant, with a learned diffusion model. We first describe a general framework that connects diffusion models to a sequence of posterior inference problems. Building on this framework, we then present the practical algorithm to train a diffusion model to sample from $p(x_0)$ by leveraging existing inference techniques.

Diffusion model from a posterior inference perspective. Consider the forward SDE in Equation (1) with the initial distribution set to the target $p(x_0)$. To draw samples from $p(x_0)$, we aim to simulate from the corresponding reverse SDE in Equation (2), which requires computing the intractable score function $\nabla_{x_t} \log p(x_t)$.

A key observation is that the intermediate variable $x_t \sim p(x_t)$ can be interpreted within a latent variable model:

$$x_0 \sim p(x_0), \qquad x_t \mid x_0 \sim p(x_t \mid x_0) := \mathcal{N}(x_t \mid \alpha_t x_0, \sigma_t^2 \mathbb{I}) \tag{3}$$

where the prior is the target, and the likelihood is the transition kernel of the forward SDE with signal and noise parameters α_t, σ_t, making x_t a noisy observation of x_0.

Based on the above observation, we can relate the score function to the posterior mean via Tweedie's formula (Roberts and Tweedie, 1996):

$$\nabla_{x_t} \log p(x_t) = \frac{\alpha_t \mathbb{E}[x_0 \mid x_t] - x_t}{\sigma_t^2}. \tag{4}$$

This connection reduces the problem of estimating the score function to estimating the posterior mean $\mathbb{E}[x_0 \mid x_t]$. Due to Gaussian conditioning, posterior inference for $p(x_0 \mid x_t)$ is often more tractable than $p(x_0)$. In particular, when $t \to 0$, the posterior is increasingly dominated by likelihood as the signal in x_t becomes stronger.

We can thus form an MC estimate of the score: we draw M approximate samples $\{x_0^m\}_{m=1}^M$ from $p(x_0 \mid x_t)$ using existing inference methods, and then estimate the score by

$$\hat{s}(x_t) := \frac{\alpha_t \hat{x}_0 - x_t}{\sigma_t^2} \approx \nabla_{x_t} \log p(x_t). \tag{5}$$

where $\hat{x}_0 := \frac{1}{M} \sum_{m=1}^M x_0^m$ approximates the posterior mean.

A similar perspective has been explored in Huang et al. (2023) for a specific diffusion parameterization. We generalize it to broader settings. Next, we propose a training procedure of diffusion models that amortizes the posterior inference results.

Diffusion model training by amortizing posterior inference. We train an amortized score network $s_\theta(x_t, t)$ targeting at the score estimate $\hat{s}(x_t)$ in Equation (5). We specify two components: (1) a procedure to generate training inputs (x_t, t) , and (2) a tractable posterior inference algorithm that constructs the training target $\hat{s}(x_t)$.

For (1), we ideally want x_t drawn from the target distribution $p(x_t)$ for each t. To this end we use the replay buffer strategy (Mnih et al., 2015; Akhound-Sadegh et al., 2024). We initialize the buffer with training inputs sampled from the initial diffusion model and continuously update it with new samples generated by the current model during training. When the buffer reaches its maximum capacity, the oldest samples are discarded. As training proceeds, the model approaches the target distribution and the training inputs are increasingly representative of the desired target.

For (2), while any approximate inference method can be used in principle, it must sufficiently explore the posterior to ensure good diffusion model performance. We primarily use Annealed Importance Sampling (AIS Neal, 2001) and include additional results with Hamiltonian Monte Carlo (HMC Neal, 2012) in Appendix C.

Our algorithm is summarized in Algorithm 1. Training begins with a replay buffer populated with samples from a randomly initialized diffusion model. In each epoch, a batch $\{x_0\}$ is drawn from the buffer, and noisy observations $\{x_t\}$ are generated from the likelihood $\mathcal{N}(x_t \mid \alpha_t x_0, \sigma_t^2 \mathbb{I})$ under a noising schedule $\{\alpha_t, \sigma_t\}$. We use an approximate posterior sampler (e.g. AIS or HMC) to estimate $\hat{s}(x_t)$ by Equation (5). The score network is then updated by minimizing a time-weighted reconstruction loss $w(t)\|s_\theta(x_t, t) - \hat{s}(x_t)\|^2$, where $w(t)$ is some time-weighting function. After each epoch, we update the buffer with new samples $\{x_0\}$ from the current diffusion model.

Algorithm 1: Diffusion training via amortizing posterior inference

Input: Number of epochs N, number of iterations per epochs I, batch size B, a randomly initialized score network $s_\theta(\cdot,\cdot)$, target distribution $p(x_0)$, schedule $\{\alpha_t, \sigma_t\}$, an approximate posterior sampler, time-weighting function $w(t)$

Output: Updated score network $s_\theta(\cdot,\cdot)$.

Initialize the buffer $\mathcal{B}$ with samples from the initial diffusion model

for *epoch* $n = 1, \cdots, N$ **do**

  Sample data $\{x_0^{(b)}\}_{b=1}^B$ from Buffer $\mathcal{B}$

  for *iteration* $i = 1, \cdots, I$ **do**

    Sample timesteps $\{t^b\}_{b=1}^B$ independently from Uniform$[0,1]$

    Sample noisy data $\{x_t^{(b)}\}_{b=1}^B$ independently from $\mathcal{N}(x_t^{(b)} \mid \alpha_t x_0^{(b)}, \sigma_t^2 \mathbb{I})$

    Estimate posterior mean $\hat{x}_0^{(b)}$ using the approximate posterior sampler targeting $p(x_0 \mid x_t^{(b)}) \propto p(x_0)\mathcal{N}(x_t^{(b)} \mid \alpha_t x_0, \sigma_t^2 \mathbb{I})$

    Estimate the score $\hat{s}^{(b)}(x_t, t) = \frac{\alpha_t \hat{x}_0^{(b)} - x_t^{(b)}}{\sigma_t^2}$

    Update θ by minimizing the loss $\sum_{b=1}^B w(t) \| s_\theta(x_t^{(b)}, t^{(b)}) - \hat{s}(x_t^{(b)}) \|^2$

  end

  Update the buffer $\mathcal{B} \leftarrow \left(\mathcal{B} \cup \{x_0\}\right)$ with new samples $\{x_0\}$ from the current model

end

4. Related Work

Simulation-based methods. Huang et al. (2023) is among the first to connect diffusion models and sampling by formulating the score estimation problem as a non-parametric posterior mean estimation one. They developed an MC sampling algorithm that first applies an importance sampling estimator, followed by an unadjusted Langevin algorithm correction to estimate the posterior mean. He et al. (2024b) proposed a similar algorithm based on rejection sampling that is shown to be effective in low-dimensional settings. However, these approaches require MC simulations at inference time and do not scale to high dimensions. Grenioux et al. (2024) extended this idea to general SDEs and developed practical strategies to efficiently sample from higher dimensional densities.

Learning-based methods. Similar to our objective, Akhound-Sadegh et al. (2024) also approaches the sampling problem by learning the score network of diffusion models. Unlike our approach based on posterior inference, they form an importance sampling estimator of the score, whose complexity scales exponentially with dimensions (Chatterjee and Diaconis, 2018). Consequently, the training objective exhibits high variance (Noble et al., 2024). Alternatively, He et al. (2024a) trains an implicit generative model that produces samples in a single step, unlike diffusion models, which rely on a multi-step sampling process. Despite this difference, it also involves solving a score function estimation problem, and therefore we consider it as a diffusion-based method. In Appendix B we show this method can be interpreted as a distilled version of our diffusion model. In addition to regressing a score network against an estimated objective, recent works also explore variational methods

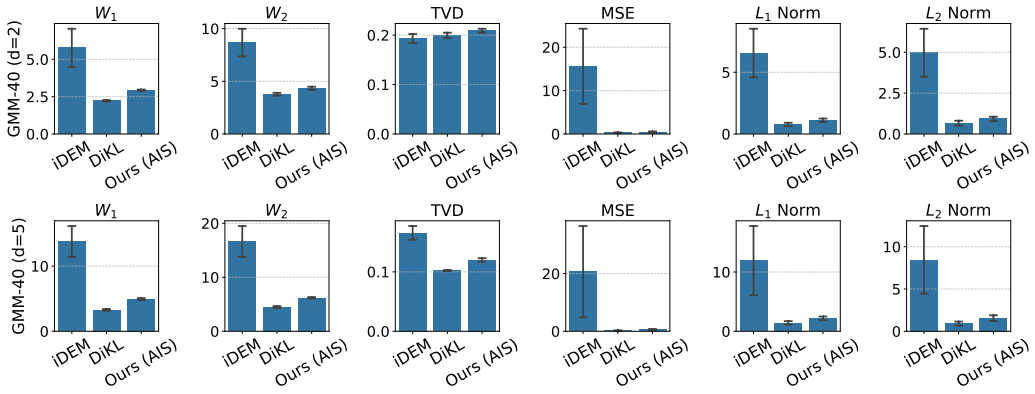

Figure 1: Comparison on the GMM-40 target ($d = 2$, top row; $d = 5$, bottom row). Results averaged over 5 runs, with error bars showing one standard error. DiKL outperforms other methods across all metrics, while our method closely follows DiKL and outperforms iDEM in both average performance and variability.

based on divergences between path measures of time-reversed diffusion processes (e.g. Zhang and Chen, 2021; Vargas et al., 2023a,b). We refer to Richter and Berner (2023) for a comprehensive review.

5. Experiments

We evaluate our method with AIS as posterior sampler on 40-component Gaussian Mixture Models (GMM-40) in both 2-dimensional and 5-dimensional settings, and a 4-particle double-well system (DW-4) in 8 dimensions. We compare our approach to two diffusion-based methods: DiKL (He et al., 2024a) and iDEM (Akhound-Sadegh et al., 2024). We report metrics including Wasserstein-1 distance ($\mathcal{W}_1$), Wasserstein-2 distance ($\mathcal{W}_2$), total variation distance (TVD), mean squared error (MSE), L_1 norm, and L_2 norm. For hyperparameter tuning, we use 2,000 target samples as validation data and choose the model with the lowest $\mathcal{W}_2$. All additional details and results are included in Appendix C.

5.1. GMM-40

We first consider the 2-dimensional GMM-40 target proposed by Midgley et al. (2023). Additionally, we include a 5-dimensional GMM target that preserves a lower dimensional mean structure. Both our method and iDEM use the same multilayer perceptron (MLP) architecture with sinusoidal positional embeddings for the score network, while DiKL employs a larger MLP for the one-step generator network.

In Figure 1, we observe that DiKL achieves the best overall performance, and our method performs comparably across all metrics. iDEM exhibits higher variance and worse average performance, particularly in W_2, MSE, and L_2 norm. See Table 1 for full results.

We note that iDEM applied a data scaling factor of 50 for GMM experiments as suggested in Akhound-Sadegh et al. (2024). In Appendix C.4, we preliminarily explore the necessity of this scaling and observe that, without proper scaling, iDEM struggles to accurately capture the structure of GMM-40.

We also experimented with our method using HMC as the posterior sampler and observed slightly worse performance than AIS, highlighting the importance of posterior inference quality. See details in Appendix C.3.

5.2. DW-4

Köhler et al. (2020) introduced the 8-dimensional DW-4 target distribution for a 4-particle system in a 2-dimensional space, which exhibits intrinsic symmetry (see details in Appendix C.5). To incorporate this symmetry, we employ the same E(3)-equivariant graph neural networks (EGNN) architecture for the score network in both our method and iDEM, while DiKL employs a larger EGNN for its one-step generator network.

Figure 2 shows that our method consistently outperforms iDEM and DiKL across most metrics, except for Energy TVD, where iDEM achieves the best result. Additionally, DiKL's one-step generator struggles to capture the more complicated density of DW-4, particularly in $\mathcal{W}_1, \mathcal{W}_2$ and Energy TVD, likely due to its limited capacity compared to multi-step approaches taken by our method and iDEM. These results highlight the potential of our method to scale to more challenging distributions where one-step generators and existing diffusion models may fall short.

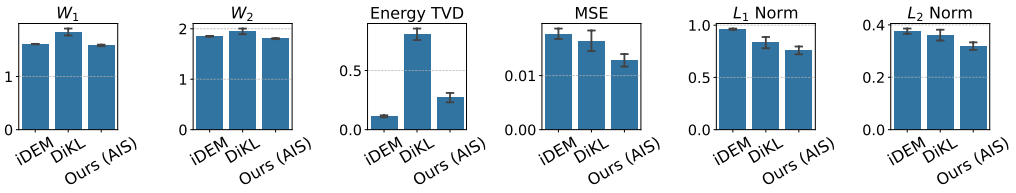

Figure 2: Comparison on the DW-4 target ($d = 8$). Results averaged over 5 runs, with error bars showing one standard error. Our method outperforms or matches iDEM and DiKL across all metrics except Energy TVD, where iDEM performs better.

6. Discussions

We introduce a diffusion-based framework to sample from unnormalized distributions by amortizing posterior inference results. A key strength of our framework is its flexibility to incorporate existing posterior samplers. In this work, we experiment with AIS and HMC as posterior samplers. Empirical results demonstrate that our method achieves competitive performance to other learning-based diffusion samplers across several benchmark targets.

The performance of our method depends on the accuracy and scalability of posterior inference methods, particularly in higher-dimensional settings. To improve the overall sampling performance, future work will explore more effective posterior inference algorithms and tuning strategies. One promising direction is to leverage the SDE parameterization and discretization strategy proposed in Grenioux et al. (2024) to mitigate the challenges of posterior inference at high noise levels. Additionally, incorporating variational inference techniques as an alternative to MC simulations for posterior inference could offer a more efficient and scalable solution to high-dimensional targets.

References

Tara Akhound-Sadegh, Jarrid Rector-Brooks, Avishek 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. *arXiv preprint arXiv:2402.06121*, 2024.

Stephen Brooks. Markov chain monte carlo method and its application. *Journal of the royal statistical society: series D (the Statistician)*, 47(1):69–100, 1998.

Sourav Chatterjee and Persi Diaconis. The sample size required in importance sampling. *The Annals of Applied Probability*, 28(2):1099–1135, 2018.

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

Rémi Flamary, Nicolas Courty, Alexandre Gramfort, Mokhtar Z. Alaya, Aurélie Boisbunon, Stanislas Chambon, Laetitia Chapel, Adrien Corenflos, Kilian Fatras, Nemo Fournier, Léo Gautheron, Nathalie T.H. Gayraud, Hicham Janati, Alain Rakotomamonjy, Ievgen Redko, Antoine Rolet, Antony Schutz, Vivien Seguy, Danica J. Sutherland, Romain Tavenard, Alexander Tong, and Titouan Vayer. Pot: Python optimal transport. *Journal of Machine Learning Research*, 22(78):1–8, 2021. URL http://jmlr.org/papers/v22/20-451.html.

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

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*, 2024a.

Ye He, Kevin Rojas, and Molei Tao. Zeroth-order sampling methods for non-log-concave distributions: Alleviating metastability by denoising diffusion. *arXiv preprint arXiv:2402.17886*, 2024b.

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

Emiel Hoogeboom, Vıctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. In *International conference on machine learning*, pages 8867–8887. PMLR, 2022.

Xunpeng Huang, Hanze Dong, HAO Yifan, Yian Ma, and Tong Zhang. Reverse diffusion monte carlo. In *The Twelfth International Conference on Learning Representations*, 2023.

Leon Klein, Andreas Krämer, and Frank Noe. Equivariant flow matching. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=eLH2NFOO1B.

Jonas Köhler, Leon Klein, and Frank Noé. Equivariant flows: Exact likelihood generative learning for symmetric densities, 2020. URL https://arxiv.org/abs/2006.02425.

Laurence Illing Midgley, Vincent Stimper, Gregor N. C. Simm, Bernhard Schölkopf, and José Miguel Hernández-Lobato. Flow annealed importance sampling bootstrap. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=XCTVFJwS9LJ.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

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

Radford M Neal. Mcmc using hamiltonian dynamics. *arXiv preprint arXiv:1206.1901*, 2012.

Maxence Noble, Louis Grenioux, Marylou Gabrié, and Alain Oliviero Durmus. Learned reference-based diffusion sampling for multi-modal distributions. *arXiv preprint arXiv:2410.19449*, 2024.

Lorenz Richter and Julius Berner. Improved sampling via learned diffusions. *arXiv preprint arXiv:2307.01198*, 2023.

Gareth O Roberts and Richard L Tweedie. Geometric convergence and central limit theorems for multidimensional hastings and metropolis algorithms. *Biometrika*, 83(1):95–110, 1996.

Tim Salimans, Thomas Mensink, Jonathan Heek, and Emiel Hoogeboom. Multistep distillation of diffusion models via moment matching. *arXiv preprint arXiv:2406.04103*, 2024.

Simo Särkkä and Arno Solin. *Applied stochastic differential equations*, volume 10. Cambridge University Press, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Surya T Tokdar and Robert E Kass. Importance sampling: a review. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(1):54–60, 2010.

Francisco Vargas, Andrius Ovsianas, David Fernandes, Mark Girolami, Neil D Lawrence, and Nikolas Nüsken. Bayesian learning via neural schrödinger–föllmer flows. *Statistics and Computing*, 33(1):3, 2023a.

Francisco Vargas, Teodora Reu, and Anna Kerekes. Expressiveness remarks for denoising diffusion based sampling. In *Fifth Symposium on Advances in Approximate Bayesian Inference*, 2023b.

Qinsheng Zhang and Yongxin Chen. Path integral sampler: a stochastic control approach for sampling. *arXiv preprint arXiv:2111.15141*, 2021.

Appendix A. Review of diffusion noising schedules

Variance-preserving (VP) noising schedule. In VP schedule (Ho et al., 2020), the coefficients of the forward SDE in Equation (1) are given by

$$f(t) = -\frac{1}{2}b(t), \quad g(t) = \sqrt{b(t)},$$

where $b(t)$ is a time-dependent function

$$b(t) = b_{\min} + t(b_{\max} - b_{\min}),$$

for some constants $b_{\min} \approx 0 \ll b_{\max}$. For a large value of $b_{\max}$, the reference distribution is approximately standard normal, $p(x_1) \approx \mathcal{N}(x_1 \mid 0, 1)$.

By Ito's calculus (see e.g. Särkkä and Solin (2019)), one can show that the transition kernel $p(x_t \mid x_0)$ in (3) are parameterized with

$$\alpha_t = \exp\left(-\frac{1}{2}\int_0^t b(s)ds\right), \quad \sigma_t^2 = \int_0^t b(s)ds.$$

Variance-exploding (VE) noising schedule. In VE schedule (Song et al., 2020), the coefficients are given by

$$f(t) = 0, \qquad g(t) = \sqrt{\frac{\mathrm{d}\sigma^2(t)}{\mathrm{d}t}},$$

for some non-negative function $\sigma(t)$ monotone in t with $\sigma(0) = 0$. For a large value of $\sigma(1)$, the reference distribution can be approximated by $p(x_1) \approx \mathcal{N}(x_1 \mid 0, \sigma(1)^2)$.

Similarly to the VP case, one can derive the following parameters of the transition kernel $p(x_t \mid x_0)$:

$$\alpha_t = 1, \qquad \sigma_t = \sigma(t).$$

Appendix B. Connection between our method and He et al. (2024a)

We establish the connection between our method and DiKL proposed by He et al. (2024a), viewing the later as a distilled version of ours.

Background on DiKL. DiKL trains a one-step generative model $p_\theta(x)$ that is induced by the following generative process,

$$z \sim \mathcal{N}(z \mid 0, \mathbb{I}), \qquad x = g_\theta(z) \tag{6}$$

where g_θ is a deterministic neural network.

The training objective for g_θ is defined as follows

$$\mathcal{L}(\theta) := \int_{t_{\min}}^{t_{\max}} w(t)\mathrm{KL}\left[p_\theta(x_t) \| p(x_t)\right] dt, \tag{7}$$

where $w(t)$ is a time-weighting function, $0 \leq t_{\min} < t_{\max} \leq 1$ are some constants, and

$$p_\theta(x_t) := \int p_\theta(x_0)\mathcal{N}(x_t \mid \alpha_t x_0, \sigma_t^2)dx_0, \tag{8}$$

$$p(x_t) := \int p(x_0)\mathcal{N}(x_t \mid \alpha_t x_0, \sigma_t^2)dx_0, \tag{9}$$

are distributions of the model and the target convolved with a Gaussian noising kernel, respectively, with $\{\alpha_t, \sigma_t\}$ some noising schedule similarly to the one that appears in diffusion models.

The gradient of the training objective can be expressed as

$$\nabla_\theta \mathcal{L} = \int w(t) \int p_\theta(x_t) \left(\nabla_{x_t} \log p_\theta(x_t) - \nabla_{x_t} \log p(x_t)\right) \frac{\partial x_t}{\partial \theta} dx_t dt. \tag{10}$$

This gradient expression involves two intractable terms, the intermediate model score $\nabla_{x_t} \log p_\theta(x_t)$, and the intermediate target score $\nabla_{x_t} \log p(x_t)$. Notably, $\nabla_{x_t} \log p(x_t)$ is exactly the score function in the diffusion reverse SDE from Equation (2) that we aim to estimate.

DiKL approaches the estimation of the score function $\nabla_{x_t} \log p(x_t)$ using similarly to ours, constructing an estimator based on posterior samples from $p(x_0 \mid x_t)$ (see their Section 4.2). To estimate the intermediate model score $\nabla_{x_t} \log p_\theta(x_t)$, they learn an auxiliary score network $s_\phi(x_t, t)$ using the denoising score matching loss (Song et al., 2020) based on samples drawn from $p_\theta(x_0)$, as in conventional diffusion training.

DiKL viewed as a distilled version of our method. In diffusion distillation literature (e.g. Salimans et al., 2024), the goal is to train a one-step generator as in Equation (6) that distills a pretraind diffusion model. The distillation objective is similar to Equation (7) where $p(x_t)$ is replaced with the distribution induced by the pretrained diffusion model. Consequently. $\nabla_{x_t} \log p(x_t)$ is directly provided by the pretrained score network.

Drawing on this connection, we can view DiKL as a distilled version of our model. This perspective aligns with the comparisons in the distillation literature, where the distilled model (DiKL) enables faster generation, while the full model (ours) remains more expressive and provides tractable density estimation.

Appendix C. Experiment Details

We evaluate our method on GMM-40 and DW-4 targets. This section details the experimental design, including diffusion setups, network architectures, posterior samplers, and validation strategies.

C.1. Parametrization Strategies

We consider two parameterizations for the score network s_θ. In our experiments, we use the noise parametrization for GMM-40 in both 2D and 5D and use the mean parametrization for DW-4.

Mean Parametrization. We learn a neural network $x_\theta(x_t, t)$ to approximate $\mathbb{E}[x_0 \mid x_t]$ and relate it to the score function via

$$\mathbb{E}[x_0 \mid x_t] = \frac{\sigma_t^2 \nabla_{x_t} \log p(x_t) + x_t}{\alpha_t}.$$

Therefore, the resulting score network is

$$s_\theta(x_t, t) := \frac{\alpha_t x_\theta(x_t, t) - x_t}{\sigma_t^2}.$$

Noise Parametrization. For a given x_0 and x_t, we define a noise variable:

$$\epsilon = \frac{x_t - \alpha_t x_0}{\sigma_t}.$$

In this case, we alternatively learn a neural network $\epsilon_\theta(x_t, t)$ to approximate $\mathbb{E}[\epsilon \mid x_t]$. Consequently, the score network is given by

$$s_\theta(x_t, t) := -\alpha_t \epsilon_\theta(x_t, t)$$

This reparametrization simplifies the learning process by focusing on the noise space, which often exhibits smaller variance and leads to more stable training.

C.2. Reported Metrics

We evaluate the performance of our models using several statistical metrics that capture different aspects of distributional similarity and sample quality. Below, we detail the key metrics employed in our experiments.

1-Wasserstein Distance ($\mathcal{W}_1$) We use the 1-Wasserstein distance to measure the discrepancy between empirical samples from the sampler and the true samples. The 1-Wasserstein distance is defined as

$$\mathcal{W}_1(\mu, \nu) = \inf_\pi \int \pi(x, y) d(x, y)\, dx\, dy, \tag{11}$$

where π is the transport plan with marginals constrained to μ and ν respectively. We use the Hungarian algorithm as implemented in the Python optimal transport package (POT) (Flamary et al., 2021).

2-Wasserstein Distance ($\mathcal{W}_2$) The 2-Wasserstein distance is defined as

$$\mathcal{W}_1(\mu, \nu) = \left(\inf_\pi \int \pi(x, y) d(x, y)\, dx\, dy \right)^{1/2}, \tag{12}$$

where π is the transport plan with marginals constrained to μ and ν respectively. We also use the Hungarian algorithm as implemented in the Python optimal transport package (POT) (Flamary et al., 2021). It is reported in Akhound-Sadegh et al. (2024); He et al. (2024a,b).

Total Variation Distance (TVD) Total Variation Distance measures the maximum difference between the probabilities assigned by two distributions over all possible events. In practice, we approximate TVD by discretizing the sample space into 1000 bins between $[-100, 100]$ in each dimension and comparing the normalized histograms of the true and model-generated samples:

$$\text{TVD} = \frac{1}{2} \sum_i |\hat{p}_i - \hat{q}_i|,$$

where $\hat{p}_i$ and $\hat{q}_i$ represent the probability mass in the i-th bin for the true and generated distributions, respectively. TVD is also reported in Akhound-Sadegh et al. (2024).

Mean Squared Error (MSE) MSE is a distance-based metric that quantifies the average of the squared differences between the mean of the predicted and true values. It is defined as:

$$\text{MSE} = \frac{1}{d} \sum_{i=1}^{d} (\overline{x}_i - \overline{\widehat{x}}_i)^2,$$

where $\overline{x}_i$ is the i-th dimension of the mean of the true samples, $\overline{\widehat{x}}_i$ is the i-th dimension of the mean of the model-generated samples and d is the dimension.

L_1 Norm The L_1 norm measures the absolute difference between the means of the true and generated samples:

$$L_1 = \|\overline{x} - \overline{\widehat{x}}\|_1 = \sum_{i=1}^{d} |\overline{x}_i - \overline{\widehat{x}}_i|,$$

where $\overline{x}_i$ is the i-th dimension of the mean of the true samples, $\overline{\widehat{x}}_i$ is the i-th dimension of the mean of the model-generated samples and d is the dimension.

L_2 Norm The L_2 norm measures the root of the sum of squared differences between the mean of the predicted and true values:

$$L_2 = \|\overline{x} - \overline{\widehat{x}}\|_2 = \sqrt{\sum_{i=1}^{d} (\overline{x}_i - \overline{\widehat{x}}_i)^2},$$

where $\overline{x}_i$ is the i-th dimension of the mean of the true samples, $\overline{\widehat{x}}_i$ is the i-th dimension of the mean of the model-generated samples and d is the dimension.

C.3. Mixture of 40 Gaussians (GMM-40)

Implementation Details. We evaluate our approach on the GMM-40 dataset as proposed in Midgley et al. (2023). The diffusion process is configured with 100 steps using a Variance Preserving (VP) noising scheme (Ho et al., 2020), with $b_{\min} = 0.1$ and $b_{\max} = 20$. The weighting function is set as the inverse signal, $w(t) = 1/\alpha_t$, to emphasize low-noise regions. The score network s_θ is implemented as a 3-layer multilayer perceptron (MLP) with a hidden dimension of 128 and SiLU activation functions. The training uses the Adam optimizer for 1000 epochs, with 100 iterations per epoch, a learning rate of 5×10^{-4}, a batch size of 64, and gradient clipping at a norm of 10.0.

For AIS-based posterior sampling, we employ Hamiltonian dynamics as the transition kernel, performing a single leapfrog step with a step size of 1.0. From 500 AIS-generated samples, one is resampled based on AIS weights and used to initialize five MALA steps with a step size of 0.01. For HMC-based posterior sampling, we generate 10 samples through five HMC iterations, each consisting of five leapfrog steps with a step size of 0.2. Hyperparameters are tuned empirically for efficient posterior exploration.

For DiKL (He et al., 2024a) and iDEM (Akhound-Sadegh et al., 2024), we use the exact setups as described in their respective papers.

We use the same validation procedure for all methods where we generate 2,000 samples using the target distribution as validation data and save the model with the lowest W_2.

For the 5-dimensional GMM-40 target, we retain the same mean structure in the first two dimensions as in the 2D case, fixing the remaining dimensions at zero. We keep the same hyperparameters and training procedures with the 2D experiments across iDEM, DiKL, and our methods to ensure comparability.

Additional Results. The scatter plots in Figure 3 visualize the samples generated from DiKL, iDEM and our methods using AIS and HMC posterior samplers compared to the ground truth samples.

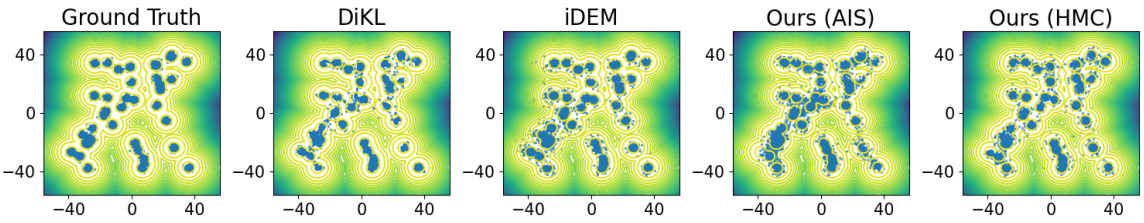

Figure 3: Scatter plot for 4000 samples from DiKL, iDEM and our methods using AIS and HMC posterior samplers from GMM-40 in 2D.

Moreover, we apply importance sampling (IS) to refine samples generated from our model. Given a set of samples, we compute the importance weights by

$$\log w_i = \log p_0(x_i) - \log p_{\text{model}}(x_i),$$

where we compute the model likelihood p_{model} by integrating the corresponding probability flow ODE (Song et al., 2020). To ensure numerical stability, we subtract the maximum log weight before exponentiation. The raw importance weights are then exponentiated and normalized:

$$\tilde{w}_i = \frac{\exp(\log w_i)}{\sum_j \exp(\log w_j)}.$$

We perform resampling with replacement according to the normalized weights $\tilde{w}_i$, producing a corrected set of samples that better represent the target distribution. Figure 4 shows that IS effectively refines the sample boundaries around the modes compared to Figure 3 for our methods. It highlights that the density estimation provided by our method can be leveraged to potentially reduce estimation bias through importance sampling.

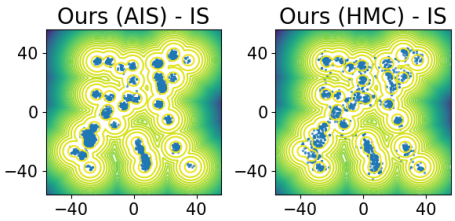

Figure 4: Scatter plot for 4000 samples from our methods using AIS and HMC posterior samplers after importance sampling.

Table 1 summarizes the performance metrics for GMM-40 in both 2D and 5D settings reported in Figure 1.

Table 1: Performance Summary for GMM-40. Metrics are reported as mean (standard error) averaged over 5 runs. DiKL achieves the best performance in all metrics and our method using AIS as the posterior sampler has close performance. Bold via Welch's two sample t-test $p < 0.1$.

Target	Metric	iDEM	DiKL	Ours (AIS)	Ours (HMC)
GMM-40 ($d = 2$)	W_1	5.77 (1.27)	**2.22 (0.05)**	2.94 (0.04)	5.69 (0.23)
	W_2	8.68 (1.31)	**3.78 (0.13)**	4.34 (0.15)	7.88 (0.28)
	TVD	**0.19 (0.01)**	**0.20 (0.01)**	0.21 (0.00)	0.21 (0.00)
	MSE	15.60 (8.67)	**0.27 (0.10)**	0.46 (0.13)	0.75 (0.17)
	L_1 Norm	6.56 (1.97)	**0.77 (0.14)**	1.11 (0.14)	1.54 (0.16)
	L_2 Norm	4.97 (1.47)	**0.67 (0.15)**	0.92 (0.14)	1.19 (0.15)
GMM-40 ($d = 5$)	W_1	13.78 (2.37)	**3.29 (0.13)**	4.94 (0.14)	6.45 (0.03)
	W_2	16.63 (2.86)	**4.48 (0.18)**	6.19 (0.12)	8.23 (0.05)
	TVD	0.17 (0.01)	**0.10 (0.00)**	0.12 (0.00)	0.11 (0.00)
	MSE	20.67 (15.82)	**0.21 (0.12)**	0.57 (0.22)	0.99 (0.08)
	L_1 Norm	11.92 (5.84)	**1.42 (0.29)**	2.18 (0.31)	3.23 (0.18)
	L_2 Norm	8.45 (4.00)	**0.91 (0.24)**	1.55 (0.34)	2.21 (0.09)

C.4. Necessity of Scaling in iDEM in GMM experiments

iDEM (Akhound-Sadegh et al., 2024) applied an additional scaling step in the GMM-40 experiments. where they scale the data by 50. We explore the necessity of scaling in iDEM for the GMM-40 (d=2). To assess its impact, we follow the exact implementation in Akhound-Sadegh et al. (2024) and change the data normalization factor to 1. Without proper scaling, iDEM struggles to accurately capture the structure of the Gaussian Mixture Model (GMM), leading to sample collapse, as shown in Figure 5 (left). In contrast, when a

scaling factor of 50 is applied, iDEM better separates the mixture components, producing a more accurate representation of the underlying distribution as shown in Figure 5 (right).

While the rationale for scaling is not disclosed in the original paper, we hypothesize that it helps reduce the separation between modes, which in turn mitigates the high variance in iDEM's estimates of intermediate scores, particularly at higher noise levels.

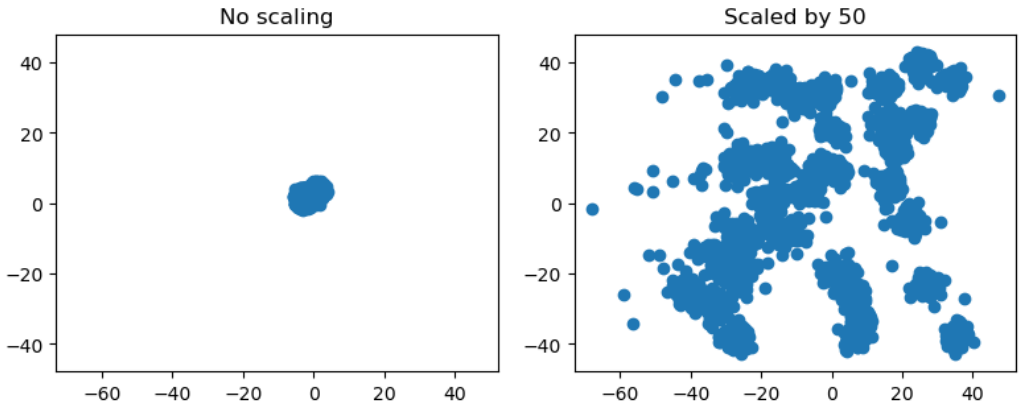

Figure 5: Comparison of scatter plots for samples from GMMs with 40 modes in 2D, with and without data scaling.

C.5. Double-Well-4 (DW-4)

DW-4 target density. The DW4 target, introduced by Köhler et al. (2020), represents the energy function of a system of 4 particles in a 2-dimensional space. The system is governed by a double-well potential, which depends on the pairwise distances of the particles. For a system of 4 particles, $x = \{x_1, \cdots, x_4\}$, the energy is given by

$$E(x) = \frac{1}{2\tau} a(d_{ij} - d_0) + b(d_{ij} - d_0)^2 + c(d_{ij} - d_0)^4, \tag{13}$$

where $d_{ij} = \|x_i - x_j\|^2$. Following previous works, we set $a = 0, b = 04, c = 0.9, \tau = 1$. The DW-4 target density is then given by $p(x) \propto \exp\{-E(x)\}$. We use a validation and test set from the MCMC samples in Klein et al. (2023) as the approximately ground-truth samples.

Implementation Details. We train our model with 100 diffusion steps with a VP noising scheme (Ho et al., 2020), setting $b_{\text{min}} = 0.1$ and $b_{\text{max}} = 20$. We apply constant weighting function and mean parametrization. The score network s_θ is designed as an E(3)-equivariant graph neural network (EGNN) following Hoogeboom et al. (2022), with three layers, a hidden dimension of 128, and SiLU activations. Training employs the Adam optimizer for 100 epochs, 100 iterations per epoch, a learning rate of 10^{-4}, a batch size of 256, and gradient clipping at a norm of 10.

AIS-based posterior sampling uses 20 importance samples and 10 AIS steps, each incorporating a one-step MALA transition with a step size of 0.01. From 1,000 AIS samples, one

is resampled according to AIS weights and refined through 50 MALA steps. The MALA step size is dynamically adjusted to maintain an acceptance rate between 0.5 and 0.6, increasing or decreasing by a factor of 1.5 based on the acceptance rate.

For DiKL (He et al., 2024a) and iDEM (Akhound-Sadegh et al., 2024), we use the exact setups as described in their respective papers.

For validation of all methods, we generate 2,000 samples using the target distribution as validation data. We save the model with the lowest W_2 across all methods.

Additional Results. Table 2 summarizes the performance metrics for DW-4 reported in Figure 2.

Table 2: Performance Summary for DW-4. Metrics are reported as mean (standard error) averaged over 5 runs. Our method outperforms or matches iDEM and DiKL in all metrics, except Energy TVD. Bold via Welch's two sample t-test $p < 0.1$

Target	Metric	iDEM	DiKL	Ours (AIS)
DW-4 ($d = 8$)	W_1	**1.61 (0.00)**	1.84 (0.06)	**1.59 (0.01)**
	W_2	1.85 (0.01)	1.95 (0.06)	**1.80 (0.01)**
	Energy TVD	**0.11 (0.01)**	0.81 (0.05)	0.27 (0.04)
	MSE	0.02 (0.00)	0.02 (0.00)	**0.01 (0.00)**
	L_1 Norm	0.96 (0.01)	**0.83 (0.05)**	**0.76 (0.04)**
	L_2 Norm	0.38 (0.01)	**0.36 (0.02)**	**0.32 (0.01)**