On scalable and efficient training of diffusion samplers

Minkyu Kim[1*] **Kiyoung Seong**[1] **Dongyeop Woo**[1] **Sungsoo Ahn**[1] **Minsu Kim**[1,2]

[1]Korea Advanced Institute of Science and Technology (KAIST) [2]Mila - Quebec AI Institute

Abstract

We address the challenge of training diffusion models to sample from unnormalized energy distributions in the absence of data, the so-called *diffusion samplers*. Although these approaches have shown promise, they struggle to scale in more demanding scenarios where energy evaluations are expensive and the sampling space is high-dimensional. To address this limitation, we propose a scalable and sample-efficient framework that properly harmonizes the powerful classical sampling method and the diffusion sampler. Specifically, we utilize Monte Carlo Markov chain (MCMC) samplers with a novelty-based auxiliary energy as a Searcher to collect off-policy samples, using an auxiliary energy function to compensate for exploring modes the diffusion sampler rarely visits. These off-policy samples are then combined with on-policy data to train the diffusion sampler, thereby expanding its coverage of the energy landscape. Furthermore, we identify primacy bias, i.e., the preference of samplers for early experience during training, as the main cause of mode collapse during training, and introduce a periodic re-initialization trick to resolve this issue. Our method significantly improves sample efficiency on standard benchmarks for diffusion samplers and also excels at higher-dimensional problems and real-world molecular conformer generation.

1 Introduction

Inference in unnormalized densities is a central challenge in machine learning, underlying probabilistic deep learning [18, 24] and many scientific applications [32, 9]. Traditionally, Markov chain Monte Carlo (MCMC) methods have been used, most prominently Metropolis-adjusted Langevin algorithms (MALA) [36] and Hamiltonian Monte Carlo (HMC) [15], but they incur repeated energy-gradient evaluations per sample. Amortized inference instead trains deep generative models to map noise to samples, enabling evaluation-free generation at test time and promising orders-of-magnitude speedups once the model is trained.

Researchers have recently focused on diffusion samplers, which parameterize continuous-time diffusion processes with neural networks, an approach inspired by successes in high-dimensional settings like image and text generation. The leading methods include flow-annealed importance sampling bootstrap (FAB) [29], generative flow networks (GFlowNets) [3], denoising diffusion samplers (DDS) [41], controlled Monte Carlo diffusion (CMCD) [42], and iterative denoising energy matching (iDEM) [1]. Because samples from the target distribution are unavailable, these samplers iterate between: (1) sample from the neural diffusion model, (2) query the energy, and (3) update the model to better match the target distribution.

Despite their promise, diffusion-based samplers struggle in high dimensions. Early in training, the neural proposal is effectively random and is not aligned with the energy landscape, leading to sample-inefficient exploration. This is in contrast with the classic training-free samplers, e.g., MALA, which leverage gradient information to steer proposals toward low-energy modes from the start.

*Correspondence to: minkyu-kim@kaist.ac.kr

Preprint. Under review.

Techniques to improve the sample efficiency of diffusion samplers, like replay buffers [3] and local energy-guided refinements [20], yield only marginal gains and fail to overcome the poor quality of the initial diffusion samples. Indeed, He et al. [17] recently showed that nearly all effective neural samplers rely on Langevin parametrization, i.e., incorporating energy gradients at inference, which erodes the primary efficiency benefit of amortized sampling.

Moreover, diffusion samplers are prone to mode collapse: training on their own outputs leads to overfitting to dominant modes, and the model "locks in" prematurely. Reinforcement learning exploration bonuses [35] can broaden coverage, but at the cost of biasing the sampler's target distribution. Local perturbations [20] help, but require many expensive iterations in large state spaces.

Contribution. We propose search-guided diffusion samplers (*SGDS*), a simple yet powerful framework that enables scalable and unbiased training of diffusion samplers in high-dimensional problems. A training-free Markov-chain "Searcher" explores the target density augmented with an explicit exploration reward to discover underexplored modes. The diffusion "Learner" then distills these trajectories through the trajectory balance objective [27], preserving theoretical guarantees while incorporating exploration.

At a high level, our *SGDS* operates in two stages. **Stage 1**: the Searcher collects informative samples from the target (optionally with exploration incentives) to overcome the random initialization of the Learner. The Learner is trained off-policy via trajectory balance on a mixture of Searcher- and self-generated trajectories, rapidly improving sample efficiency. **Stage 2**: the Searcher employs random network distillation (RND) bonuses [10] to probe modes the Learner has not yet covered; the Learner then ingests these enriched trajectories using trajectory balance with weight re-initialization to counter primacy bias [31].

We show that *SGDS*, despite its simplicity, produces substantial gains over baseline diffusion samplers across benchmarks: classical Gaussian mixtures and the Manywell task; particle simulation problems like LJ-13 and LJ-55; and a real-world molecule, Alanine Dipeptide. Our method significantly improves sample efficiency and scalability, marking a practical path towards high-dimensional diffusion-based inference.

2 Preliminaries

2.1 Diffusion samplers as controlled neural SDEs

Let $\mathcal{E}\colon \mathbb{R}^d \to \mathbb{R}$ be an energy function defining an unnormalized density $R(x) = \exp\big(-\mathcal{E}(x)\big)$. Sampling from the corresponding Boltzmann distribution $p_{\text{target}}(x) = R(x)/Z$, with partition function $Z = \int R(x)\,dx$, can be formulated as controlling the stochastic differential equation (SDE)

$$\mathrm{d}x_t = u_\theta(x_t, t)\,\mathrm{d}t + g(x_t, t)\,\mathrm{d}w_t, \qquad x_0 \sim \mu_0,\ t \in [0, 1], \tag{1}$$

where w_t is standard d-dimensional Brownian motion, u_θ is the drift function parameterized by θ e.g., neural networks, and g is the diffusion function. The goal is to choose θ such that the terminal distribution p_1^θ induced by Equation (1) matches the target, i.e., $p_1^\theta(x) \propto R(x)$).

Euler–Maruyama discretization. With T uniform steps of size $\Delta t := 1/T$, the SDE Equation (1) is discretized via the Euler–Maruyama scheme

$$x_{t+\Delta t} = x_t + u_\theta(x_t, t)\,\Delta t + g(x_t, t)\sqrt{\Delta t}\,z_t, \qquad z_t \sim \mathcal{N}(0, I_d), \tag{2}$$

which defines Gaussian forward kernels $P_F(x_{t+\Delta t} \mid x_t; \theta)$. Analogously, one defines reference backward kernels $P_B(x_{t-\Delta t} \mid x_t)$. Common choices for P_B include Brownian motion $\mathrm{d}x_t = \beta(t)\,\mathrm{d}\bar{w}_t$ for variance-exploding (VE) processes, the time-reversed Ornstein–Uhlenbeck (OU) kernel $\mathrm{d}x_t = -\beta(t)x_t\,\mathrm{d}t + \sqrt{2\beta(t)}\,\mathrm{d}\bar{w}_t$ for variance-preserving (VP) processes, and the Brownian bridge $\mathrm{d}x_t = \frac{x_t}{t}\mathrm{d}t + \sigma\mathrm{d}\bar{w}_t$, where $\bar{w}_t$ is time-reversed Brownian motion.

The forward and backward policies for the complete trajectory $\tau = (x_0 \to x_{\Delta t} \to \cdots \to x_1)$, denoted by $P_F(\tau; \theta)$ and $P_B(\tau \mid x_1)$, repectively, are defined as compositions of these kernels across discrete time steps:

$$P_F(\tau; \theta) = \prod_{i=0}^{T-1} P_F\big(x_{(i+1)\Delta t} \mid x_{i\Delta t}; \theta\big), \quad P_B(\tau \mid x_1) = \prod_{i=0}^{T-1} P_B\big(x_{(i-1)\Delta t} \mid x_{i\Delta t}\big). \tag{3}$$

Algorithm 1 Training search-guided diffusion samplers (*SGDS*)

$\mathcal{Q}_{\text{buffer}} \leftarrow \emptyset$; **fix** random target net f_{rnd}; initialize predictor $\hat{f}_\phi$ and Learner $(P_F(\tau;\theta), \log Z_\theta)$
for $r = 1, \ldots, N_{\text{round}}$ **do** ▷ *outer rounds*
  // Searcher: gradient-guided MCMC
  $\tilde{\mathcal{E}}(x) \leftarrow \begin{cases} \mathcal{E}(x), & r = 1, \\ \mathcal{E}(x) - \alpha \left\| f_{\text{rnd}}(x) - \hat{f}_\phi(x) \right\|_2^2, & r > 1 \end{cases}$
  Obtain $\{x_1^{(i)}\}_{i=1}^{M_{\text{chain}}}$ and $\log \hat{Z}$ by running M_{chain} parallel MCMC for M_{iter} steps on $\tilde{\mathcal{E}}(x)$
  $\mathcal{Q}_{\text{buffer}} \leftarrow \mathcal{Q}_{\text{buffer}} \cup \{x_1^{(i)}, \mathcal{E}(x_1^{(i)})\}_{i=1}^{M_{\text{chain}}}$
  // Learner: I inner iterations (even iterations: on-policy, odd iterations: off-policy)
  for $i = 1, \ldots, I$ **do**
    if $i \bmod 2 = 0$ **then** ▷ *on-policy*
      Sample $\{\tau_k\}_{k=1}^{B} \sim P_F(\tau;\theta)$
      $\mathcal{X} \leftarrow \{x_1 \text{ from } \tau_k\}$
    else ▷ *off-policy*
      Sample $\mathcal{X} = \{x_1\}_{\ell=1}^{B_{\text{off}}} \sim P(\cdot \mid \mathcal{Q}_{\text{buffer}})$
      Generate $\{\tau_\ell\} \sim P_B(\tau \mid x_1)$
    end if
    $\mathcal{L}_{\text{TB}} = \frac{1}{B} \sum_k \left[\log \frac{Z_\theta P_F(\tau_k;\theta)}{R(x_1) P_B(\tau_k \mid x_1)}\right]^2$
    $\theta \leftarrow \text{Minimize}(\mathcal{L}_{\text{TB}})$ ▷ *diffusion sampler update*
    $\phi \leftarrow \text{Minimize}\left(\frac{1}{|\mathcal{X}|} \sum_{x_1 \in \mathcal{X}} \| f_{\text{rnd}}(x_1) - \hat{f}_\phi(x_1) \|_2^2\right)$ ▷ *RND predictor update*
  end for
  Re-initialize $P_F(\cdot \mid \theta)$ but retain $\log Z_\theta$ ▷ *Periodic partial re-initialization*
end for

Stochastic control of neural SDEs. Diffusion models typically minimize the forward Kullback–Leibler (KL) divergence

$$D_{\text{KL}}\big(P_B(\tau \mid x_1)\, p_{\text{target}}(x_1) \,\|\, P_F(\tau;\theta)\, \mu_0(x_0)\big),$$

which presupposes abundant samples from $x_1 \sim p_{\text{target}}$. When such data are unavailable, e.g., in scientific domains, one may instead minimize the reverse KL divergence

$$D_{\text{KL}}\big(P_F(\tau;\theta)\, \mu_0(x_0) \,\|\, P_B(\tau \mid x_1)\, p_{\text{target}}(x_1)\big),$$

using samples from $x_1 \sim P_F$. Notable methods that optimize this objective include the path-integral sampler (PIS) [45], which employs a VE Brownian-motion reference process, and denoising diffusion samplers (DDS) [41], which use a VP OU reference process.

2.2 Continuous GFlowNet objective for diffusion samplers

Following Sendera et al. [38], Euler–Maruyama samplers can be interpreted as continuous generative flow networks (GFlowNets) [23]. GFlowNets [3, 4] are off-policy reinforcement-learning algorithms for sequential decision making samplers. Treating the initial state x_0 as a point mass at the origin, the forward policy P_F acts as an agent that sequentially constructs a trajectory τ. The trajectory balance (TB) criterion [27] guarantees that the density induced by P_F matches the target distribution:

$$Z_\theta\, P_F(\tau;\theta) = R(x_1)\, P_B(\tau \mid x_1), \qquad \forall \tau, \tag{4}$$

where Z_θ is a learnable scalar that approximates the unknown partition function Z. Existing GFlowNet-based samplers [44, 38] often adopt Brownian-bridge kernels for P_B.

Applying the TB condition to sub-trajectory of τ yields the *sub-trajectory balance* objective [26, 34, 44]. While this variant can improve credit assignment, it estimates marginal densities at intermediate states with higher bias compared to the global TB estimates [38].

Off-policy property of GFlowNet-based diffusion samplers. In contrast to KL-based objectives such as PIS or DDS, using on-policy training, GFlowNet objectives can be optimized with *off-policy* trajectories drawn from any proposal distribution with full support. This flexibility enables richer exploration strategies—noisy roll-outs [23], replay buffers, and MCMC-based local search [38]—that are crucial for efficient sampling from multimodal distributions.

3 Method

3.1 Search-guided diffusion samplers (*SGDS*): overall framework

In this section, we describe the overall framework of the search-guided diffusion samplers (*SGDS*). Our *SGDS* combines the strengths of *off-policy* training from GFlowNet diffusion samplers with the exploratory power of gradient-guided MCMC. We follow the setting of Sendera et al. [38] for modeling GFlowNet-based diffusion samplers. Each *round* alternates between two roles:

Searcher (gradient-informed MCMC). The Searcher uses gradient information $\nabla \log \pi(x)$ to efficiently generate representative samples from the target distribution. These samples populate a replay buffer and simultaneously provide an estimate of the log partition function, $\log Z$. Exploration is guided by an intrinsic reward from random network distillation (RND) [10], which identifies underexplored modes using a form of self-supervised learning.

Learner (diffusion sampler). Learner, a neural diffusion sampler, is trained by minimizing trajectory balance loss [23], blending (i) *on-policy* trajectories generated from its current policy and (ii) *off-policy* trajectories replayed from the buffer. Periodic re-initialization of the Learner mitigates primacy bias, enhancing sample efficiency.

This round repeats until the Learner alone generates high-quality samples. For simple targets, training may converge within a single round, while complex targets typically benefit from multiple rounds.

The *SGDS* tackles two critical challenges in existing diffusion sampling approaches:

Scalability. In high-dimensional spaces, diffusion samplers frequently miss low-energy modes, as their generated samples rarely visit unexplored modes. The Searcher, operating as parallel gradient-informed chains, rapidly identifies these modes. Although the samples collected from the Searcher are biased, the trajectory balance objective enables unbiased training of the Learner.

Sample efficiency. Each expensive gradient evaluation is amortized across multiple Learner updates through off-policy replay. The RND-driven intrinsic rewards direct the Searcher towards under-explored areas, maximizing the informativeness of new samples. Periodic Learner re-initialization prevents overfitting to initial samples and maintains replay buffer diversity. Collectively, these components significantly enhance the efficiency of gradient computations.

Algorithmic details for each component follow in subsequent sections and Algorithm 1.

3.2 Searcher

The Searcher identifies low-energy modes using parallel gradient-guided Markov chains. Methods such as annealed importance sampling (AIS) [30], Metropolis-adjusted Langevin algorithms (MALA) [36], or molecular dynamics (MD) are suitable candidates. These methods generate samples by transporting prior samples in the direction of the target density (or its tempered density) via several Markov chains. We use AIS and MALA for synthetic energy functions, and MD for all-atom systems.

In the initial step of the algorithm, we run M_{chain} parallel chains, estimating $\log \hat{Z}$ which is explained in Appendix A. The Searcher then stores the collected samples in a replay buffer and passes the estimated $\log \hat{Z}$ to the Learner model. In subsequent rounds, we incorporate exploration uncertainty from the Learner via intrinsic rewards for exploration, modifying the Searcher's energy landscape as:

$$\tilde{\mathcal{E}}(x) = \mathcal{E}(x) - \alpha \log r_{\text{intrinsic}}(x). \tag{5}$$

Here, $r_{\text{intrinsic}}(x)$ highlights underexplored modes based on previous Learner experiences, and the gradient is used in the drift function of SDEs. Adding a repulsive term for exploration resembles the core idea of metadynamics, which biases sampling away from the modes that have already been well captured.

Random network distillation (RND). To efficiently guide exploration, we employ RND [10] to quantify state novelty, steering the Searcher towards underexplored consists of a fixed, randomly initialized network $f(x)$ and a trainable predictor network $\hat{f}(x;\phi)$ trained by minimizing:

$$\mathcal{L}_{\text{RND}}(x) = \|f(x) - \hat{f}(x;\phi)\|_2^2, \tag{6}$$

and, for the Searcher in the next round, we utilize this loss as the intrinsic reward given by:

$$r_{\text{intrinsic}}(x) = \exp(\|f(x) - \hat{f}(x;\phi)\|_2^2). \tag{7}$$

High prediction errors indicate novel states. RND training uses replay buffer samples and online trajectories, assigning high novelty to underexplored modes.

3.3 Learner

With the replay buffer initialized by Searcher's samples, the Learner minimizes the trajectory balance objective through iterative training, combining online and replay trajectories. The training incorporates:

$$\mathcal{L}_{\text{off-policy}}(\theta) = \mathbb{E}_{\tau\sim P_B(\tau|x_1),x_1\sim P(x_1|\mathcal{D}_{\text{buffer}})}\frac{1}{2}\left[\log\frac{Z_\theta P_F(\tau;\theta)}{R(x_1)\,P_B(\tau\mid x_1)}\right]^2, \tag{8}$$

$$\mathcal{L}_{\text{on-policy}}(\theta) = \mathbb{E}_{\tau\sim P_F(\tau)}\frac{1}{2}\left[\log\frac{Z_\theta P_F(\tau;\theta)}{R(x_1)\,P_B(\tau\mid x_1)}\right]^2. \tag{9}$$

Here $P(x_1 \mid \mathcal{D}_{\text{buffer}})$ denotes a *rank-based* sampling distribution [40] that assigns higher probability to lower energy samples stored in the buffer, focusing replay on promising modes.

We leverage both on-policy and off-policy training signals from online trajectories and replayed samples, with a replay ratio γ determining the frequency of replay updates (default: $\gamma = 1$).

Re-initialization. Learner re-initialization mitigates primacy bias commonly observed in reinforcement learning scenarios. Primacy bias [31] refers to the model's tendency to rely excessively on early experiences, being trapped in low-reward or biased samples generated at initial stages, thereby hindering the discovery of high-reward samples and underexplored modes. Periodically re-initializing the Learner model $P_F(\cdot|\theta)$ alleviates this bias by resetting parameters strongly influenced by early samples, allowing faster adaptation to recent, higher-quality experiences. Crucially, we retain the previously learned $\log Z_\theta$ parameter and the replay buffer, preserving the accumulated knowledge while allowing the network to recalibrate based on updated experiences.

4 Related works

Classical samplers. Classical sampling approaches primarily rely on MCMC methods. This includes gradient-based algorithms like MALA [36] and HMC [15]. Annealing-based techniques, such as AIS [30] and SMC [12], introduce intermediate distributions to gradually approximate complex targets, mitigating mode collapse. While these MCMC-based methods enable sampling from the complex unnormalized density, they require long trajectories and extensive energy evaluations.

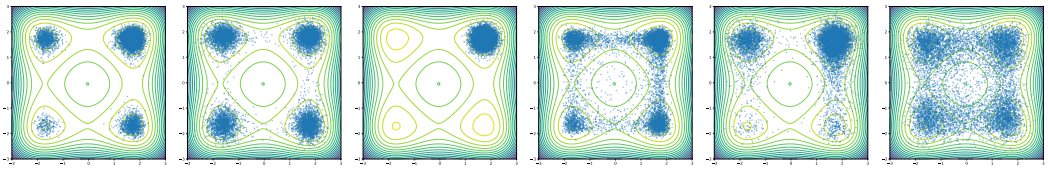
Neural amortized inference. Neural amortized inference methods aim to bypass costly MCMC by training neural samplers that generate approximate samples in one or a few forward passes. Diffusion-based neural samplers learn stochastic differential equations parameterized by neural networks to map simple priors to complex targets [45, 41], and GFlowNets train stochastic policies whose marginal visitation probabilities match an unnormalized density [3, 13]. Boltzmann Generators (BG) is another line of works to amortize inference, such as molecular dynamics simulation. BG utilizes normalizing flows trained on simulated data to sample from the Boltzmann distribution and estimate density, enabling statistical reweighting for unbiased estimates [33, 14, 29, 22, 39].

Diffusion-based neural samplers. Diffusion-based samplers aim to sample from unnormalized target distributions in data-free settings. Several approaches [45, 41, 42, 2, 5] formulate the sampling objective via KL divergence in path measure space. Akhound-Sadegh et al. [1] further introduces off-policy training via replay buffers. Recent works [8, 11] also explore controllable dynamics, offering improved exploration in complex energy landscapes. While these methods often improve mode coverage by learning reverse-time dynamics, they remain computationally intensive, hindering scalability in high-dimensional settings.

Generative Flow Networks. GFlowNets was originally introduced by Bengio et al. [3] and Bengio et al. [4] on discrete spaces where the probability of each outcome is proportional to a given reward signal. Subsequent extensions have connected GFlowNets to continuous space [23], enabling sampling from unnormalized densities in high-dimensional spaces [13, 28]. Recent work has also explored enhancements to off-policy training strategies [38] and incorporated local search mechanisms [20], allowing GFlowNets to more effectively navigate continuous energy landscapes. Additionally, adaptive reward design has emerged as a promising direction for improving mode coverage during training [21], especially in tasks that require structured exploration or sparse supervision.

Table 1: ELBO, EUBO, their gap, and energy calls across high-dimensional Manywell distributions. We use MALA as the local search algorithm. We consume 6M energy calls per searcher (12M total for 2 rounds) and 8M energy calls for the learner. **Bold** indicates the best performance per metric, and * indicates large absolute values of metrics.

	Manywell ($d = 64$)				**Manywell** ($d = 128$)			
Method	ELBO ↑	EUBO ↓	EUBO − ELBO ↓	Energy calls	ELBO ↑	EUBO ↓	EUBO − ELBO ↓	Energy calls
PIS+LP	300.57 ± 0.37	347.48 ± 0.26	46.91 ± 0.55	130M	601.01 ± 0.94	697.32 ± 0.49	96.31 ± 0.71	130M
TB+LP	306.47 ± 0.23	351.98 ± 0.46	45.52 ± 0.51	180M	612.45 ± 0.65	706.73 ± 2.59	94.28 ± 3.00	300M
FL-SubTB+LP	306.14 ± 0.71	352.22 ± 0.62	46.08 ± 0.26	330M	609.85 ± 0.48	709.96 ± 2.10	99.61 ± 1.83	330M
TB+LS+LP	312.66 ± 2.66	339.34 ± 1.02	26.68 ± 3.37	320M	592.52 ± 2.25	693.65 ± 1.40	101.81 ± 3.62	320M
TB+Expl+LP	306.54 ± 0.23	351.91 ± 0.53	45.37 ± 0.66	180M	611.98 ± 0.34	705.35 ± 1.05	93.37 ± 1.22	240M
TB+Expl+LS+LP	300.10 ± 1.05	344.85 ± 0.41	44.75 ± 1.39	320M	591.47 ± 0.36	694.93 ± 0.54	103.45 ± 0.88	320M
PIS	**321.87 ± 0.05**	2026.11 ± 408.98	1704.91 ± 408.49	100M	**643.30 ± 0.09**	1159.60 ± 48.53	516.30 ± 49.67	100M
TB	317.35 ± 6.01	853.94 ± 43.35	544.36 ± 29.85	100M	637.01 ± 2.14	1423.35 ± 292.15	786.35 ± 290.46	100M
TB+LS	314.94 ± 4.60	357.40 ± 4.36	42.91 ± 9.15	290M	573.13 ± 73.49	738.07 ± 10.77	164.95 ± 62.71	290M
TB+Expl+LS	265.99 ± 95.39	361.00 ± 16.58	41.46 ± 15.47	290M	589.49 ± 7.25	698.24 ± 2.81	108.74 ± 10.07	290M
GAFN	320.88 ± 0.36	573.68 ± 29.02	252.80 ± 30.87	100M	*	*	*	100M
AT + LP	281.56 ± 2.21	353.64 ± 3.48	72.48 ± 2.97	370M	462.61 ± 6.67	739.93 ± 4.97	277.32 ± 2.46	370M
iDEM	268.99 ± 1.22	414.18 ± 1.06	145.20 ± 1.60	300M	494.28 ± 2.94	817.32 ± 3.22	323.04 ± 5.69	300M
SGDS	320.25 ± 0.13	**336.51 ± 0.11**	**16.26 ± 0.22**	**20M**	614.41 ± 3.44	**684.76 ± 1.30**	**70.35 ± 4.31**	**20M**



(a) Ground Truth (b) *SGDS* (c) PIS (d) SubTB+LP (e) TB+Expl+LS (f) iDEM

Figure 1: Mode coverage comparison using 2D projections of 10,000 samples on Manywell-128.

Connection to previous works. Using gradient-guided MCMC for improving exploration in off-policy diffusion samplers is not new. Lemos et al. [25] employed gradient-guided MCMC to populate replay buffers for GFlowNet diffusion sampler training. Sendera et al. [38] applied parallel MALA initialized from diffusion sampler states, similar to discrete local search GFlowNet methods [20]. Our approach extends the multiple-round algorithm of Lemos et al. [25], incorporating RL techniques to boost efficiency. It can be viewed as a deeper but shorter-cycle alternative to Sendera et al. [38], whose frequent diffusion-based re-initializations overly depend on sampler performance (see comparison with TB + LS at Table 1, Table 2, and Figure 4a).

Leveraging Learner uncertainty to guide exploration aligns with active learning and related GFlowNet approaches [35, 21]. Following generative augmented flow network (GAFN) [35], direct injection of intrinsic reward was effective, similar to our idea (see comparison with GAFN at Table 1). While Kim et al. [21] introduced additional neural samplers called adaptive teachers (AT) as Searchers to covers high loss region it is highly unstable in large scale due to Searcher's adversarial behavior with non-stationary objective, where our method efficiently employs MCMC-based exploration without additional neural network (see comparison with AT + LP at Table 1).

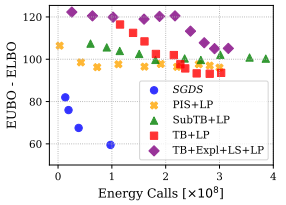
5 Experiments

In this section,[1] our primary goal is to demonstrate the performance and efficiency of our proposed framework through several experiments. Specifically, we aim to showcase the sample efficiency and scalability of our method, as well as validate the effectiveness of the various training strategies we introduced. We focus on presenting results on high-dimensional tasks. In all the experiments, we use four different random seeds and average the results of each run. We provide the full experimental details and additional results in Appendix A.

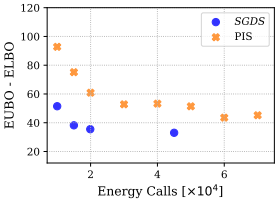
5.1 Main results

Settings. In this work, we compare the performance of our proposed framework against baselines on multiple benchmark tasks, including 40GMM, Manywell-32/64/128, LJ-13, and LJ-55. We

[1] Source code: https://anonymous.4open.science/r/SGDS-D38C



(a) Manywell-128



(b) LJ-55

Method	ELBO ↑	EUBO ↓
AIS+MLE	244.85 ± 10.34	840.45 ± 11.59
Fine-tuning	554.17 ± 107.70	707.07 ± 8.49
Re-init w/o log Z	583.65 ± 16.69	691.37 ± 1.40
w/o RND-searcher	608.01 ± 4.18	686.63 ± 1.32
SGDS	**614.41 ± 3.44**	**684.76 ± 1.30**

(c) Ablation results on Manywell-128

Figure 2: Trade-off between EUBO–ELBO gap and energy calls in Manywell-128 (left) and LJ-55 (middle). The results of ablation study on Manywell-128 (right) show the performance of AIS using the same total energy calls with MLE amortizing, taking 2 rounds with fine-tuning instead of re-initialization, and using the Searcher with no RND rewards. All methods use 20M energy calls.

Table 2: ELBO, $\widehat{\text{EUBO}}$, their gap, and energy calls across Lennard-Jones potential. We denote $\widehat{\text{EUBO}}$ as the EUBO metrics calculated by the reference samples provided by [1], which are not exact samples from the target distribution. **Bold** indicates the best performance, and * indicates large absolute values of metrics.

	LJ-13 ($d = 39$)				**LJ-55** ($d = 165$)			
Method	ELBO ↑	$\widehat{\text{EUBO}}$ ↓	$\widehat{\text{EUBO}}$ - ELBO ↓	Energy calls	ELBO ↑	$\widehat{\text{EUBO}}$ ↓	$\widehat{\text{EUBO}}$ - ELBO ↓	Energy calls
PIS	**57.73 ± 0.22**	59.77 ± 0.23	2.04 ± 0.32	370K	357.19 ± 3.67	410.15 ± 4.90	45.53 ± 5.58	45K
TB	54.73 ± 3.02	67.26 ± 1.63	12.53 ± 3.43	370K	*	*	*	45K
TB+Expl+LS	52.82 ± 0.30	64.81 ± 0.42	11.99 ± 0.52	3M	*	563.81 ± 23.26	*	1M
iDEM	27.88 ± 5.92	140.25 ± 4.81	112.37 ± 7.63	300M	*	*	*	120M
SGDS	57.68 ± 0.19	**59.21 ± 0.16**	**1.53 ± 0.25**	370K	**363.22 ± 0.87**	**396.23 ± 0.33**	**33.01 ± 0.93**	45K

evaluate methods using three metrics: the Evidence Lower Bound (ELBO), Evidence Upper Bound (EUBO) [7], and the EUBO – ELBO gap. A smaller gap between ELBO and EUBO indicates a more accurate approximation of the target distribution.

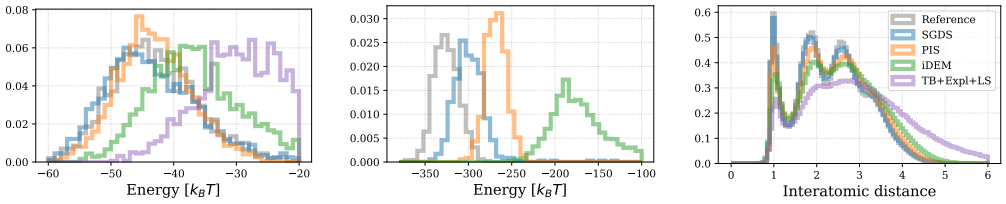
For fair comparison on the number of energy calls, we train the methods until convergence of ELBO and EUBO. To determine convergence, we evaluate based on the moving average of the metrics over the 10 consecutive evaluations, where we evaluate the model every 100 training steps. If a method does not converge within the maximum number of epochs, we report the metrics at the final step.

Baselines. The baselines are primarily selected based on their strong performance demonstrated in the prior work [38], as well as their methodological relevance [35, 21] or having a different framework [1]. Specifically, iDEM [1] utilizes trajectories of length $T = 1,000$ for SDE integration, whereas other baselines, including PIS [45], TB [27], AT [21], and GAFN [35], employ shorter diffusion trajectories ($T = 100$) with distinct optimization objectives. We further evaluate enhanced variants of these methods incorporating LP, such as PIS+LP, TB+LP, and FL-SubTB+LP, along with exploration-enhanced (+Expl) or local search (+LS) variants introduced by Sendera et al. [38].

For the LJ potentials, we omit LP-based methods due to their poor convergence despite using additional information. We also note that our comparison with iDEM is based on our reimplementation, where we modified the hyperparameters to improve sample efficiency (see the details in Appendix A).

Results. As shown in Table 1 and Table 2, our proposed framework consistently achieves superior performance across all high-dimensional tasks (Manywell-64, Manywell-128, and LJ-55). Especially, our method demonstrates the best trade-off between performance and energy efficiency, using a small number of energy calls.

In Figure 1, one can observe that our method better captures the modes in Manywell-128 when compared to the baselines. As illustrated in Figure 2a and Figure 2b, even increasing the energy budget of baselines does not allow them to surpass the performance of our proposed approach. Also, as shown in Figure 3, our framework generates high quality samples with low energy. Furthermore, for the LJ-55 potential, the distribution of interatomic distances is similar to the ground truth distribution. Additionally, our method obtains competitive results with significantly fewer samples in lower-dimensional tasks such as 40GMM, Manywell-32 (see Appendix B), and LJ-13 (see Table 2).



(a) LJ-13 energy histogram (b) LJ-55 energy histogram (c) LJ-55 interatomic distance

Figure 3: Histograms for LJ-13/55 energy densities and LJ-55 interatomic distances.

5.2 Ablation study

MCMC sampler with the same budget. In our method, we consume energy calls during both Searcher sampling and Learner training. To evaluate the efficiency of the Learner's on/off-policy mixing training scheme, we conduct a controlled comparison where the total energy call budget (20M) is entirely allocated to AIS on Manywell-128. We run 200 chains on the trajectories with $T = 10,000$ for AIS. As a result, even though high-reward samples were collected using AIS with much longer trajectories, the MLE Learner failed to perform amortized learning as shown in Figure 2c.

Periodic re-initialization and pre-trained flow. We perform an ablation study to evaluate two design components of our method when proceeding to the next training round: (1) re-initializing the Learner model, and (2) retaining the pre-trained $\log Z$ parameter from the previous round. Specifically, to assess the benefit of re-initialization in mitigating primacy bias, we compare our method against a fine-tuning baseline where the second-round Learner continues training from the first-round model weights without re-initialization. To isolate the effect of retaining the estimated $\log Z$ value, we compare against a variant where the $\log Z$ parameter is also re-initialized at the start of the second round. As shown in Figure 2c, our full method outperforms both ablation variants, confirming that re-initialization is beneficial for mitigating primacy bias, and that employing the $\log Z$ parameter across rounds leads to better training stability and performance.

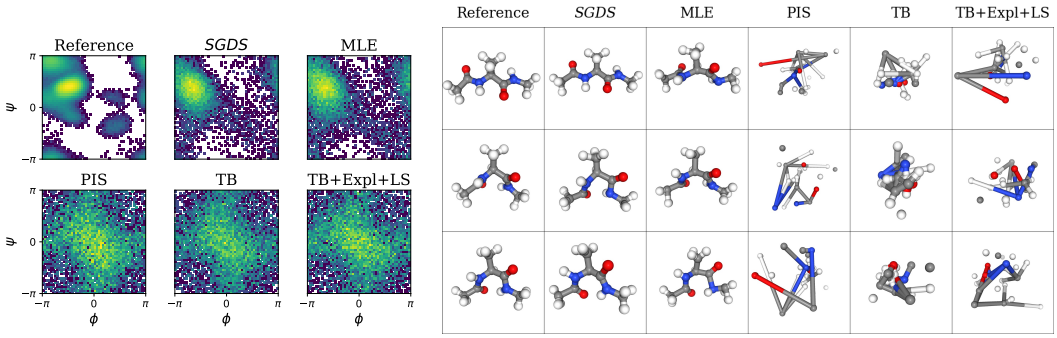
Novelty-based reward in Searcher sampling. We assess the effectiveness of incorporating the novelty-based intrinsic reward derived by RND [10] into the Searcher sampling process in later training rounds. In our framework, starting from the second round, the Searcher sampler drives prior samples in the direction of the target distribution and exploration signal derived from a previously trained RND module, which prioritizes underexplored modes by the Learner sampler. These dynamics guide the Searcher to focus sampling efforts on modes that remain novel and close to the target distribution across rounds. As shown in Figure 2c, at the end of round 2, our RND-augmented approach yields a smaller EUBO–ELBO gap compared to a way of repeating the same Searcher sampling without exploration. These results demonstrate that using intrinsic rewards to adaptively bias Searcher sampling toward novel modes improves overall distributional coverage across rounds.

5.3 Application to molecular conformer generation.

We also consider a real-world system, Alanine Dipeptide (ALDP), consisting of 22 atoms in vacuum at a temperature of 300K. While some previous works show promising results in sampling its conformation, they rely on low-dimensional descriptors such as rotatable torsion angles [43]. Solving ALDP at all-atom resolutions remains a challenge for existing diffusion-based neural samplers.

Settings. To accurately evaluate molecular energies, we employ TorchANI [16], a PyTorch implementation of ANI deep learning potentials trained on quantum-mechanical reference data. For the Searcher, we run four parallel 55ps Langevin dynamics simulations under the TorchANI potential. In the first round, simulations are performed at 600 K to efficiently sample slow degrees of freedom; in the second round, we use 300 K to capture faster motions and collect high-reward samples. The Learner and RND models use the $E(3)$-equivariant graph neural network (EGNN) architecture [37] based on atomic coordinates. We provide details in Appendix A.

Baselines. To establish a baseline, we generated a reference-state ensemble via a 100 ns Langevin-dynamics simulation at 300 K. We evaluated three methods: PIS, TB, and a local-search variant of TB that employs the same Langevin dynamics as the Searcher. We compared SGDS to maximum-



(a) Ramachandran plots

(b) 3D visualization of sampled conformations

Figure 4: Qualitative results of methods in Alanine Dipeptide. (a) Ramachandran plot consisting of two backbone torsion angles (ϕ, ψ) and (b) 3D visualization of generated conformations, respectively.

likelihood estimation (MLE) of forward path distribution, training MLE on 2.5 times more samples, generated by the same Searcher without RND, using 10 parallel simulations to match the total number of energy evaluations. We omit the LP methods since they have large absolute values of EUBO and ELBO. We exclude comparison with Volokhova et al. [43], as they consider only rotatable torsion angles, and with Midgley et al. [29], which employs a discrete normalizing flow on internal coordinates, whereas our method utilizes a diffusion model in atomic coordinate space.

Results. In Table 3, our method outperforms diffusion-based neural samplers and MLE. As illustrated in Figure 4, both our method and MLE capture the free energy landscape and generate physically plausible conformations by leveraging high-fidelity samples from the Langevin dynamics Searcher. By contrast, diffusion-based neural samplers (PIS, TB, and TB+Expl+LS) fail to reconstruct the target free-energy surface or to produce realistic structures, since their forward policies insufficiently explore the complex landscape. We note that the Langevin dynamics used in the Searcher yields higher-quality samples than those obtained by local searches from forward-policy outputs. Furthermore, our approach refines the biased samples from the high-temperature Searcher through an unbiased TB objective, improving ELBO and EUBO scores compared to MLE.

Table 3: Comparison of ELBO, EUBO, and energy calls on Alanine Dipeptide (ALDP). **Bold** indicates the best performance, and * indicates large absolute values of metrics.

Method (Energy calls)	ELBO [$\times 10^3$] ↑	EUBO [$\times 10^3$] ↓
PIS (1M)	516.912 ± 5.357	601.565 ± 87.771
TB (1M)	*	*
TB+Expl+LS (3M)	519.614 ± 0.015	538.479 ± 0.198
MLE (1M)	520.618 ± 0.142	538.032 ± 0.000
SGDS (1M)	**520.916 ± 0.165**	**538.025 ± 0.003**

6 Conclusion

We have proposed a scalable and sample-efficient sampling framework *SGDS* that integrates an MCMC Searcher with a diffusion Learner. By leveraging high-quality samples from replay buffers and training the Learner model via on/off-policy TB objectives, our method effectively bridges classical sampling with neural amortization. The inclusion of novelty-based intrinsic rewards by RND further enhances the exploration of the Searcher, enabling informed guidance to underexplored modes throughout multiple rounds.

Our work opened promising directions for integrating learning-based amortization with classical sampling, particularly for tasks where both diversity and precision are crucial. Future extensions include designing multi-agent search systems that leverage classical sampling methods for cooperative strategic exploration in high-dimensional spaces and developing advanced off-policy learning schemes, such as adaptive filtering strategies for the replay buffer.

Acknowledgements

This work was partly supported by Institute for Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (RS-2019-II190075, Artificial Intelligence Graduate School Support Program(KAIST)), National Research Foundation of Korea(NRF) grant funded by the Ministry of Science and ICT(MSIT) (No. RS-2022-NR072184), GRDC(Global Research Development Center) Cooperative Hub Program through the National Research Foundation of Korea(NRF) grant funded by the Ministry of Science and ICT(MSIT) (No. RS-2024-00436165), and the Institute of Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (RS-2025-02304967, AI Star Fellowship(KAIST)).
Minsu Kim was supported by KAIST Jang Yeong Sil Fellowship.

References

[1] 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 *International Conference on Machine Learning (ICML)*, 2024.

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

[3] Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. Flow network based generative models for non-iterative diverse candidate generation. *Neural Information Processing Systems (NeurIPS)*, 2021.

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

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

[6] Julius Berner, Lorenz Richter, Marcin Sendera, Jarrid Rector-Brooks, and Nikolay Malkin. From discrete-time policies to continuous-time diffusion samplers: Asymptotic equivalences and faster training. *arXiv preprint arXiv:2501.06148*, 2025.

[7] Denis Blessing, Xiaogang Jia, Johannes Esslinger, Francisco Vargas, and Gerhard Neumann. Beyond ELBOs: A large-scale evaluation of variational methods for sampling. In *International Conference on Machine Learning (ICML)*, 2024.

[8] Denis Blessing, Julius Berner, Lorenz Richter, and Gerhard Neumann. Underdamped diffusion bridges with applications to sampling. In *International Conference on Learning Representations (ICLR)*, 2025.

[9] Ignasi Buch, Toni Giorgino, and Gianni De Fabritiis. Complete reconstruction of an enzyme-inhibitor binding process by molecular dynamics simulations. *Proceedings of the National Academy of Sciences*, 108(25):10184–10189, 2011.

[10] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. In *International Conference on Learning Representations (ICLR)*, 2019.

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

[12] Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 68(3):411–436, 2006.

[13] Tristan Deleu, António Góis, Chris Emezue, Mansi Rankawat, Simon Lacoste-Julien, Stefan Bauer, and Yoshua Bengio. Bayesian structure learning with generative flow networks. *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2022.

[14] Manuel Dibak, Leon Klein, Andreas Krämer, and Frank Noé. Temperature steerable flows and boltzmann generators. *Physical Review Research*, 4(4):L042005, 2022.

[15] Simon Duane, A.D. Kennedy, Brian J. Pendleton, and Duncan Roweth. Hybrid Monte Carlo. *Physics Letters B*, 195(2):216–222, 1987.

[16] Xiang Gao, Farhad Ramezanghorbani, Olexandr Isayev, Justin S Smith, and Adrian E Roitberg. Torchani: a free and open source pytorch-based deep learning implementation of the ani neural network potentials. *Journal of chemical information and modeling*, 60(7):3408–3415, 2020.

[17] Jiajun He, Yuanqi Du, Francisco Vargas, Dinghuai Zhang, Shreyas Padhy, RuiKang OuYang, Carla Gomes, and José Miguel Hernández-Lobato. No trick, no treat: Pursuits and challenges towards simulation-free training of neural samplers. *arXiv preprint arXiv:2502.06685*, 2025.

[18] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

[19] Emiel Hoogeboom, Vıctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. In *International Conference on Machine Learning (ICML)*, 2022.

[20] Minsu Kim, Taeyoung Yun, Emmanuel Bengio, Dinghuai Zhang, Yoshua Bengio, Sungsoo Ahn, and Jinkyoo Park. Local search GFlowNets. *International Conference on Learning Representations (ICLR)*, 2024.

[21] Minsu Kim, Sanghyeok Choi, Taeyoung Yun, Emmanuel Bengio, Leo Feng, Jarrid Rector-Brooks, Sungsoo Ahn, Jinkyoo Park, Nikolay Malkin, and Yoshua Bengio. Adaptive teachers for amortized samplers. In *International Conference on Learning Representations (ICLR)*, 2025.

[22] Leon Klein and Frank Noé. Transferable boltzmann generators. *Neural Information Processing Systems (NeurIPS)*, 2024.

[23] Salem Lahlou, Tristan Deleu, Pablo Lemos, Dinghuai Zhang, Alexandra Volokhova, Alex Hernández-Garcıa, Léna Néhale Ezzine, Yoshua Bengio, and Nikolay Malkin. A theory of continuous generative flow networks. *International Conference on Machine Learning (ICML)*, 2023.

[24] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, Fujie Huang, et al. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

[25] Pablo Lemos, Nikolay Malkin, Will Handley, Yoshua Bengio, Yashar Hezaveh, and Laurence Perreault-Levasseur. Improving gradient-guided nested sampling for posterior inference. In *International Conference on Machine Learning (ICML)*, 2024.

[26] Kanika Madan, Jarrid Rector-Brooks, Maksym Korablyov, Emmanuel Bengio, Moksh Jain, Andrei Nica, Tom Bosc, Yoshua Bengio, and Nikolay Malkin. Learning GFlowNets from partial episodes for improved convergence and stability. *International Conference on Machine Learning (ICML)*, 2022.

[27] Nikolay Malkin, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. Trajectory balance: Improved credit assignment in gflownets. *Neural Information Processing Systems (NeurIPS)*, 2022.

[28] Nikolay Malkin, Salem Lahlou, Tristan Deleu, Xu Ji, Edward Hu, Katie Everett, Dinghuai Zhang, and Yoshua Bengio. GFlowNets and variational inference. *International Conference on Learning Representations (ICLR)*, 2023.

[29] Laurence Illing Midgley, Vincent Stimper, Gregor N. C. Simm, Bernhard Schölkopf, and José Miguel Hernández-Lobato. Flow annealed importance sampling bootstrap. In *International Conference on Learning Representations (ICLR)*, 2023.

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

[31] Evgenii Nikishin, Max Schwarzer, Pierluca D'Oro, Pierre-Luc Bacon, and Aaron Courville. The primacy bias in deep reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2022.

[32] Frank Noé, Christof Schütte, Eric Vanden-Eijnden, Lothar Reich, and Thomas R Weikl. Constructing the equilibrium ensemble of folding pathways from short off-equilibrium simulations. *Proceedings of the National Academy of Sciences*, 106(45):19011–19016, 2009.

[33] 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.

[34] Ling Pan, Nikolay Malkin, Dinghuai Zhang, and Yoshua Bengio. Better training of GFlowNets with local credit and incomplete trajectories. *International Conference on Machine Learning (ICML)*, 2023.

[35] Ling Pan, Dinghuai Zhang, Aaron Courville, Longbo Huang, and Yoshua Bengio. Generative augmented flow networks. In *International Conference on Learning Representations (ICLR)*, 2023.

[36] Peter J Rossky, Jimmie D Doll, and Harold L Friedman. Brownian dynamics as smart monte carlo simulation. *The Journal of Chemical Physics*, 69(10):4628–4633, 1978.

[37] Vıctor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E (n) equivariant graph neural networks. In *International Conference on Machine Learning (ICML)*, pages 9323–9332. PMLR, 2021.

[38] Marcin Sendera, Minsu Kim, Sarthak Mittal, Pablo Lemos, Luca Scimeca, Jarrid Rector-Brooks, Alexandre Adam, Yoshua Bengio, and Nikolay Malkin. Improved off-policy training of diffusion samplers. *Neural Information Processing Systems (NeurIPS)*, 2024.

[39] Charlie B. Tan, Joey Bose, Chen Lin, Leon Klein, Michael M. Bronstein, and Alexander Tong. Scalable equilibrium sampling with sequential boltzmann generators. In *Frontiers in Probabilistic Inference: Learning meets Sampling*, 2025.

[40] Austin Tripp, Erik Daxberger, and José Miguel Hernández-Lobato. Sample-efficient optimization in the latent space of deep generative models via weighted retraining. *Neural Information Processing Systems (NeurIPS)*, 2020.

[41] Francisco Vargas, Will Grathwohl, and Arnaud Doucet. Denoising diffusion samplers. *International Conference on Learning Representations (ICLR)*, 2023.

[42] Francisco Vargas, Shreyas Padhy, Denis Blessing, and Nikolas Nüsken. Transport meets variational inference: Controlled Monte Carlo diffusions. *International Conference on Learning Representations (ICLR)*, 2024.

[43] Alexandra Volokhova, Michał Koziarski, Alex Hernández-García, Cheng-Hao Liu, Santiago Miret, Pablo Lemos, Luca Thiede, Zichao Yan, Alán Aspuru-Guzik, and Yoshua Bengio. Towards equilibrium molecular conformation generation with gflownets. *Digital Discovery*, 3 (5):1038–1047, 2024.

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

[45] Qinsheng Zhang and Yongxin Chen. Path integral sampler: a stochastic control approach for sampling. *International Conference on Learning Representations (ICLR)*, 2022.

A Experiment details

Code is available at https://github.com/minkyu1022/SGDS.

And the reference samples can be downloaded from https://zenodo.org/records/15436773.

A.1 MCMC samplers for Searcher

Annealed importance sampling (AIS). Annealed importance sampling (AIS) [30] is an MCMC sampling method for estimating the partition functions of target distributions. AIS bridges between an easy-to-sample initial distribution $\pi_0(x)$ and a target distribution $\pi_T(x)$ through a sequence of intermediate distributions $\{\pi_t(x)\}_{t=0}^{T}$, where T is the length of a trajectory or chain. Each intermediate distribution $\pi_t(x)$ typically has the form:

$$\pi_t(x) \propto \pi_0(x)^{1-\beta_t}\pi_T(x)^{\beta_t}, \quad 0 = \beta_0 < \beta_1 < \cdots < \beta_T = 1, \tag{10}$$

where $\{\beta_t\}$ is a predefined annealing schedule, and we use $\beta_t = \frac{t}{T}$ in our framework. AIS generates samples through an MCMC transition kernel at each intermediate distribution with the following SDE simulation:

$$\mathrm{d}x_t = \nabla \log \pi_t(x_t)\mathrm{d}t + \sqrt{2}\mathrm{d}W_t, \tag{11}$$

where $\nabla \log \pi_t(x_t) = (1-\beta_t)\nabla \log \pi_0(x_t) + \beta_t \nabla \log \pi_T(x_t)$ is the score function of the annealed distribution (unnormalized). Then it accumulates importance weights given by:

$$w_{\mathrm{AIS}} = \prod_{t=1}^{T} \frac{\pi_t(x_{t-1})}{\pi_{t-1}(x_{t-1})}, \tag{12}$$

and the expectation of these weights provides an unbiased estimator of the partition function ratio between $\pi_T(x)$ and $\pi_0(x)$:

$$\frac{Z_T}{Z_0} \approx \frac{1}{N}\sum_{i=1}^{N} w^{(i)}, \tag{13}$$

where $w^{(i)}$ is the importance weight computed for the i-th AIS trajectory, and N is the total number of trajectories. We compute the unbiased estimation of the log scale of the partition function for Manywell experiments by

$$\log \hat{Z}_T = \log \frac{1}{N}\sum_{i=1}^{N} w^{(i)}, \tag{14}$$

where $\log Z_0 = 0$ because the initial distribution is Gaussian in our framework.

Metropolis-Adjusted Langevin Algorithm (MALA). The Metropolis-Adjusted Langevin Algorithm (MALA) [36] is an MCMC method that uses the gradient of the energy function to generate samples from a target distribution $\pi(x)$. MALA starts by sampling initial states $x_0 \sim \pi_0(x_0)$, where $\pi_0(\cdot)$ is some proposed initial distribution (in most cases, $\mathcal{N}(0, \sigma^2 I)$). It then iteratively proceeds transition from x_t to x_{t+1} by simulating the following Langevin dynamics:

$$\mathrm{d}x_t = -\nabla \mathcal{E}(x_t)\mathrm{d}t + \sqrt{2}\mathrm{d}W_t, \tag{15}$$

Here, x_t is the current state at time t, W_t denotes the standard Brownian motion, and $\mathcal{E}(x)$ is the energy function of target distribution $\pi(x)$, i.e. $-\nabla\mathcal{E}(x_t) = \nabla \log \pi(x_t)$.

The proposed sample x_{t+1} is then accepted or rejected according to the Metropolis-Hastings acceptance probability:

$$\alpha = \min\left\{1, \frac{\pi(x_{t+1})q(x_t \mid x_{t+1})}{\pi(x_t)q(x_{t+1} \mid x_t)}\right\}, \tag{16}$$

where $q(\cdot \mid \cdot)$ denotes the Gaussian transition density induced by the Langevin proposal:

$$x_{t+1} = x_t - \nabla\mathcal{E}(x_t)\Delta t + \sqrt{2\Delta t} \cdot z, \quad z \sim \mathcal{N}(0, I). \tag{17}$$

The step size Δt is a key factor influencing the quality of sampling. For all tasks, we utilize the scheduling of step size, by comparison between the current acceptance rate and the target acceptance rate (57.4%). We use MALA as Searcher on 40GMM, LJ-13, and LJ-55.

Also, since a MALA trajectory forms a Markov chain, consecutive samples are still correlated and therefore $\{x_i\}_{i=1}^N$ are not strictly i.i.d. To reduce the most severe correlations we discard the first $M_{\text{burn-in}}$ iterations as burn-in and use all subsequent states directly. We then compute a rough estimate

$$\log \hat{Z} = \log\Big[\frac{1}{N}\sum_{i=1}^{N} \exp\left(-\mathcal{E}(x_i)\right)\Big], \tag{18}$$

where this estimator is biased since $x_i \sim \pi$ ideally and $\mathbb{E}_\pi[\exp\left(-\mathcal{E}(x)\right)] = Z\int \pi^2(x)dx < Z$. Despite the bias, the estimation can provide a numerically reasonable heuristic value for the initialization of the Learner's $\log Z_\theta$.

Underdamped Langevin dynamics. For MCMC Searchers of a real-world molecule, Alanine Dipeptide, we adopt underdamped Langevin dynamics as our molecular dynamics (MD). This framework combines deterministic forces with stochastic fluctuations, which is essential for accurately capturing thermal motion and inertial effects of the molecules. The resulting dynamics are governed by the following system of stochastic differential equations:

$$\begin{aligned} \mathrm{d}x_t &= v_t\,\mathrm{d}t, \\ \mathrm{d}v_t &= -M^{-1}\nabla\mathcal{E}(x_t)\,\mathrm{d}t - \gamma v_t\,\mathrm{d}t + \sqrt{2\gamma k_B T\, M^{-1}}\,\mathrm{d}W_t. \end{aligned} \tag{19}$$

Here, x_t is the position at time t, v_t is the velocity, M is the mass matrix (symmetric positive definite), $\mathcal{E}(x)$ is the potential energy function, and $\nabla\mathcal{E}(x_t)$ is its gradient with respect to position, i.e., the negative force. The parameter γ is the friction coefficient, k_B is the Boltzmann constant, T is the absolute temperature, and W_t denotes standard Brownian motion.

For ALDP, we use underdamped Langevin dynamics as MD with high temperature(600K). We use Euler-Maruyama integration to discretize the Langevin dynamics. As in MALA, we compute $\log \hat{Z}$ for the initialization of $\log Z_\theta$ in Learner, using Equation (18).

A.2 Metrics

In this subsection, we formally define the evaluation metrics used to assess the Learner's quality. All metrics are derived from the same importance-weight formulation based on the target partition function.

We begin with the exact log partition function $\log Z$, which can be written using forward-path importance sampling. Let $\tau = (x_0, x_{\Delta t}, \ldots, x_1)$ denote a sample trajectory drawn from the forward policy $P_F(\tau)$, and let $R(x_1)$ be the reward associated with the final state x_1. Then, the partition function can be expressed as

$$\log Z = \log \mathbb{E}_{\tau\sim P_F(\tau)}\left(\frac{R(x_1)P_B(\tau \mid x_1)}{P_F(\tau)}\right), \tag{20}$$

where $P_B(\tau \mid x_1)$ is the backward policy conditioned on the final state.

Since directly optimizing this quantity is intractable, we use two surrogate bounds. The first is the evidence lower Bound (ELBO), defined as

$$\text{ELBO} = \mathbb{E}_{\tau\sim P_F(\tau)}\left[\log \frac{R(x_1)P_B(\tau \mid x_1)}{P_F(\tau)}\right]. \tag{21}$$

By Jensen's inequality, ELBO is always a lower bound on the true $\log Z$. It is commonly used as a training objective and can reflect how well the forward policy P_F concentrates on high-reward trajectories. However, ELBO can be misleading in practice. A high ELBO does not necessarily imply

that all important modes are captured, as the forward policy may collapse to a small subset of modes while still achieving high reward [7].

To address this limitation, we also evaluate the evidence upper Bound (EUBO), which flips the sampling distribution:

$$\text{EUBO} = \mathbb{E}_{\tau \sim P_B(\tau)} \left[\log \frac{R(x_1) P_B(\tau \mid x_1)}{P_F(\tau)} \right]. \tag{22}$$

Unlike ELBO, EUBO acts as a diagnostic metric. It is an upper bound of $\log Z$ and penalizes missing probability mass. EUBO is driven to penalize missing probability mass and therefore exposes mode-collapse that ELBO may hide [7]. And then, true $\log Z$ is consequently bounded by two bounds, i.e., $\text{ELBO} \leq \log Z \leq \text{EUBO}$.

A smaller gap between the two bounds yields a tighter estimate of $\log Z$, making this gap a useful indicator of the Learner's sampling quality.

Table 4: Searcher configurations of SGDS

Benchmark	40GMM	Manywell 32	Manywell 64	Manywell 128	LJ-13	LJ-55	ALDP
Type	MALA	AIS	AIS	AIS	MALA	MALA	MD
# of Chains	300	60K	60K	60K	16	1	4
Chain length	4K	100	100	100	4K	10K	110K
Burn-in	2K	-	-	-	2K	4K	10K
init. step size	1e-3	1e-3	1e-3	1e-3	1e-5	1e-5	0.5fs

Table 5: Learner configurations of SGDS

Benchmark	40GMM	Manywell 32	Manywell 64	Manywell 128	LJ-13	LJ-55	ALDP
Brownian bridge std (σ)	10.0	1.0	1.0	1.0	0.2	0.2	0.2
Buffer size	600k	60k	60k	60k	50K	10K	800K
Batch size	300	300	300	300	32	4	16
Architecture	MLP	MLP	MLP	MLP	EGNN	EGNN	EGNN
hidden dim	256	256	256	256	64	64	128
# of layers	2	2	2	2	5	5	5
RND weight	100	100	100	100	10	1	10

A.3 Experimental setup

For the diffusion-based neural samplers, we follow the setup of [38].

Gaussian mixture model with 40 modes (40GMM). Training proceeds in one or two rounds. Our framework achieves competitive performance against baselines even with only a single round, and shows marginal improvement with a second round. We use MALA as the Searcher, running 300 parallel chains of length 4K, discarding the first 2K steps as burn-in. We maintain a target acceptance rate of 57.4% through step size scheduling, resulting in a total of 2.4M energy evaluations. We use the Gaussian prior with a standard deviation of 21.0 for MALA.

All methods adopt the PIS architecture [45, 38], with a joint network consisting of a two-layer MLP with 256 hidden dimensions. The RND network consists of three layers in the predictor network and the target network, with 256 hidden dimensions. We adopt Brownian bridges as the backward process, with a Brownian motion coefficient of 10.0. We run 25K epochs in both the first round and the second round.

Manywell distributions. We proceed with one or two rounds for training on Manywell distributions. We use AIS as the Searcher, running 60K parallel chains (3K chains * 20 iterations) of length 100, only taking the final step samples. We use the Gaussian prior with a standard deviation of 1.0.

All methods adopt the PIS architecture [45, 38], with a joint network consisting of a two-layer MLP with 256 hidden dimensions. The RND network consists of three layers in the predictor network and the target network, with 256 hidden dimensions. We adopt Brownian bridges as the backward process, with a Brownian motion coefficient of 1.0. We run 25K epochs in the first round and 30K in the second round.

Lennard-Jones (LJ) potentials. Training proceeds in two rounds. We use MALA as the Searcher for two rounds: in LJ-13 we run 16 parallel chains of length 4K corresponding to 64K energy evaluations, discarding the first 2K steps as burn-in and retaining 57.4% accepted samples among remaining 32K samples; in LJ-55 we run a single chain of length 10K corresponding to 10K energy evaluations, discarding the first 4K steps and retaining 57.4% accepted samples among remaining 6K samples. We use the Gaussian prior with a standard deviation of 1.75 for MALA.

All methods utilize five EGNN layers with 64 hidden dimensions. Following [19, 22], we design an E(3)-equivariant generative model initialized from a Dirac delta at the origin, using a mean-free forward transition kernel in inference. The RND network comprises three layers in the predictor network and two in the target network. We adopt Brownian bridges as the backward process for diffusion-based neural samplers, with a Brownian motion coefficient of 0.2. For LJ-13, we run 5K epochs in the first round and 10K in the second round; for LJ-55, 10K and then 20K epochs.

Specifically, we note that the reported performance of the iDEM on Table 2 differs from the original paper [1] due to adjustments, except $\sigma_{\max}$ and $\sigma_{\min}$ of the noise scheduling, made to avoid significant discrepancies in energy call usage compared to our method. We reduce the EGNN hidden dimension to 64 and the batch size to 8, and limit the total number of training epochs, including both inner and outer loops, to 15K accordingly. And while the latest iDEM codebase employs 10 steps of Langevin dynamics refinement before evaluation, particularly for LJ-55, we omit this step for fair comparison and instead set the number of samples for MC estimation to 1K. While iDEM reports a lower bound of $\log Z$ computed via importance sampling with its learned proposal density $q(x)$ given by OT-CFM model, we omit this result in our tables. We compute the lower bound based on trajectory-level estimators without training auxiliary models, i.e., CFM. Thus, our reported values are not directly comparable to those from iDEM.

Additionally, in LJ-55, we maximize the log-likelihood of the forward path distribution under the backward process for the first 5K epochs of each round, discretizing backward paths from Brownian bridges initialized with empirical samples collected by Searchers. We also use randomized time scheduling introduced in [6] for our method. We train PIS at a learning rate of $1e-4$, TB at a learning rate of $2e-4$, and SGDS at a learning rate of $5e-4$. We use 4 and 32 batch sizes for all methods except PIS in LJ-13 and LJ-55, respectively. For PIS, we halve these sizes due to the memory limitation required by the forward SDE computational graph.

Alanine Dipeptide. We perform two rounds of search using under-damped Langevin dynamics. In each round, we run four parallel simulations of 55 ps each, with a time step of 0.5 fs, requiring 440K energy evaluations. We discard the first 5 ps of each trajectory as burn-in, then collect 400K samples. Each simulation starts from the same initial position drawn from a Dirac delta distribution, with all initial velocities set to zero. We integrate equations of motion using the Euler–Maruyama integrator, set the friction coefficient $\gamma = 1$, and use temperature $T = 600K$ for the first round Searcher and $T = 300K$ for the second round Searcher.

Similar to LJ potentials, all models utilize five EGNN layers with 128 hidden dimensions. We use a Dirac delta prior distribution at the origin and a mean-free forward transition kernel to guarantee $E(3)$-equivariance of the marginal density in inference. The Learner network comprises five EGNN layers, while the predictor network and target network in the RND framework contain three and two layers, respectively. As in LJ potentials, we use the Brownian motion coefficient of 0.2. We run 10K epochs in the first round and 20K epochs in the second. As in LJ-55, we maximize the log-likelihood for the first 5K epochs each round. We also utilize randomized time scheduling for our method. We train PIS at a learning rate of $1e-4$ and all other methods at $5e-4$. We use a 16 batch size for all methods except PIS, which uses an 8 batch size due to the memory limitation required by the forward SDE computational graph.

In inference time, we follow [22]. We first align the topology of generated samples with the target bond graph since the architecture and machine learning potential have a degree of freedom in atom ordering. We first match the bond graphs of generated samples with a given bond graph of interest and then correct the chirality of the generated sample to fit the target molecular configuration. The generated sample is rejected if the bond graph is not isomorphic to the target bond graph.

A.4 Task details

40-Component Gaussian Mixture Model (40GMM). The 40-component Gaussian Mixture Model (GMM) consists of a mixture distribution of 40 Gaussian components, each characterized by a distinct

Table 6: ELBO, EUBO, their gap, and Energy calls on 40GMM and Manywell-32.

	40GMM ($d = 2$)				**Manywell** ($d = 32$)			
Method	ELBO ↑	EUBO ↓	Gap ↓	Energy calls	ELBO ↑	EUBO ↓	Gap ↓	Energy calls
PIS+LP	-1.32 ± 0.07	2.42 ± 0.20	3.75 ± 0.22	300M	160.83 ± 0.41	180.49 ± 4.76	19.66 ± 4.78	300M
TB+LP	-0.35 ± 0.03	0.53 ± 0.04	0.87 ± 0.03	160M	161.42 ± 0.40	195.89 ± 8.14	34.37 ± 8.15	300M
FL-SubTB+LP	-0.36 ± 0.01	0.58 ± 0.08	0.94 ± 0.07	260M	160.74 ± 0.15	215.93 ± 4.52	55.19 ± 4.52	330M
TB+LS+LP	-0.38 ± 0.03	0.32 ± 0.02	0.69 ± 0.02	320M	162.95 ± 0.08	166.30 ± 0.11	3.35 ± 0.14	320M
TB+Expl+LP	-0.37 ± 0.01	0.32 ± 0.02	0.69 ± 0.02	300M	160.76 ± 0.13	215.92 ± 14.90	55.16 ± 14.90	300M
TB+Expl+LS+LP	-0.37 ± 0.01	0.34 ± 0.02	0.71 ± 0.02	320M	162.97 ± 0.06	166.25 ± 0.10	3.28 ± 0.12	320M
PIS	-2.03 ± 0.22	55.48 ± 10.71	57.50 ± 9.02	100M	159.71 ± 1.70	333.79 ± 3.98	174.08 ± 4.33	100M
TB	-1.35 ± 0.04	99.04 ± 6.01	100.40 ± 5.67	100M	160.58 ± 0.87	439.28 ± 166.52	278.70 ± 166.49	100M
TB+LS	-0.38 ± 0.03	0.83 ± 0.46	1.21 ± 0.38	290M	163.12 ± 0.10	166.05 ± 0.12	2.93 ± 0.16	290M
TB+Expl+LS	-0.38 ± 0.05	0.58 ± 0.34	0.96 ± 0.34	290M	160.87 ± 3.31	168.27 ± 1.49	7.40 ± 3.63	290M
GAFN	*	*	*	N/A	161.02 ± 0.05	282.40 ± 2.02	121.38 ± 2.02	100M
iDEM	-2.14 ± 0.45	12.75 ± 3.67	14.89 ± 3.70	300M	142.23 ± 0.40	211.56 ± 2.53	69.33 ± 2.56	300M
Ours (round 1)	-0.40 ± 0.01	0.33 ± 0.02	0.73 ± 0.02	**6M**	162.49 ± 0.05	166.60 ± 0.01	4.11 ± 0.05	**9M**
Ours (round 2)	-0.40 ± 0.03	0.33 ± 0.05	0.73 ± 0.05	**12M**	162.63 ± 0.01	166.48 ± 0.03	3.85 ± 0.03	**20M**

mean vector μ_i. The energy function for the GMM is defined as:

$$\mathcal{E}(x) = -\log\left(\frac{1}{n}\sum_{i=1}^{n}\mathcal{N}(x;\mu_i,\sigma^2 I)\right),$$

where $n = 40$, the weight of each i-th Gaussian component is the same, and $\mathcal{N}(x;\mu_i,\sigma^2 I)$ is the probability density function of the multivariate Gaussian distribution.

ManyWell distributions. The Manywell potential describes a high-dimensional energy landscape containing multiple wells (local minima), each representing stable states with distinct energy levels. The energy function of Manywell distribution is given by:

$$\mathcal{E}(x) = \sum_{k=1}^{n}(x_{2k-1}^4 - 6x_{2k-1}^2 - \frac{1}{2}x_{2k-1} + \frac{1}{2}x_{2k}^2) + C,$$

where $n = d/2$ is the number of wells, and d is the dimensionality of the landscape. Adjusting the dimensionality $d = 2n$ allows varying the number of wells and complexity, creating tasks like Manywell-32, Manywell-64, and Manywell-128.

Lennard-Jones (LJ) potentials. The Lennard-Jones potential models the interactions between particles. The energy function is defined as:

$$\mathcal{E}(x) = 2\kappa \sum_{1\le i<j\le N} \epsilon\left[\left(\frac{\sigma}{r_{ij}}\right)^{12} - 2\left(\frac{\sigma}{r_{ij}}\right)^{6}\right] + \frac{\lambda}{2}\sum_{i=1}^{N}\|r_i - r_{cm}\|^2, \tag{23}$$

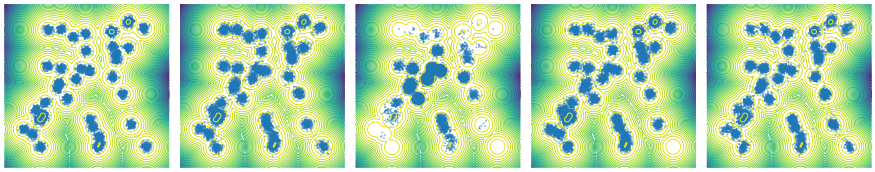
where ϵ and κ are parameters defining the depth of the potential well and the energy factor, respectively. $r_{ij} = \|x_i - x_j\|$ represents the Euclidean distance between particles i and j. σ is the characteristic distance at which the potential between two particles vanishes, often interpreted as the van der Waals radius. In our experiments, we set all parameters to 1.0, i.e., $\kappa = \epsilon = \sigma = \lambda = 1.0$. Adjusting the number of particles creates tasks such as LJ-13 and LJ-55, increasing the complexity of the particle interactions and resulting in a rugged energy landscape.

TorchANI potential for Alanine Dipeptide. We leverage TorchANI [16], a PyTorch implementation of ANI deep-learning potentials trained on quantum-mechanical reference data, to accurately calculate molecular energies. It provides transferable machine learning potential trained on organic molecules for efficient energy and force evaluations with accuracy comparable to density-functional theory (DFT). In particular, TorchANI excels at modeling small to medium-sized organic molecules such as alanine dipeptide.

B Additional experimental results

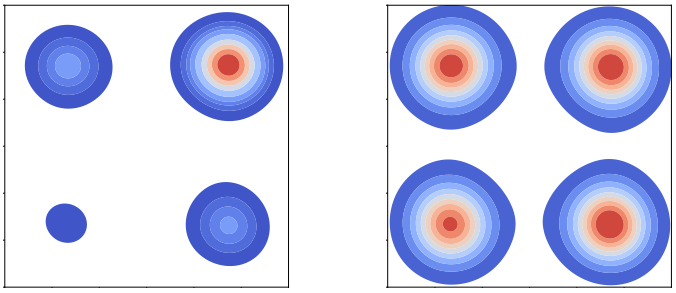
B.1 Low-dimensional standard benchmarks

Baselines and settings. We benchmark our framework on two standard low-dimensional tasks: 40GMM and Manywell-32. Consistent with the high-dimensional experiments, we report ELBO,

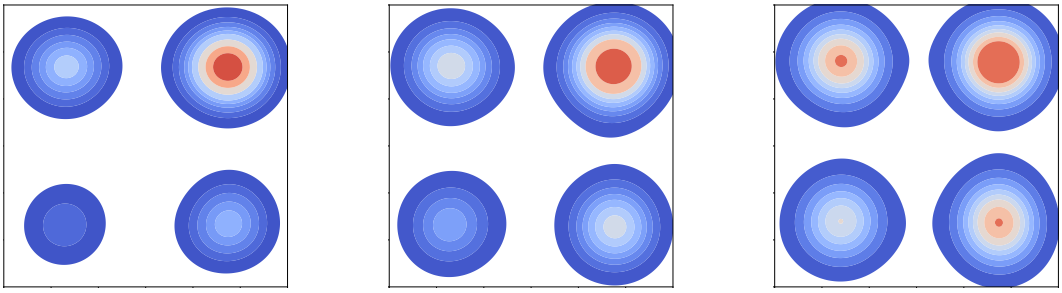


(a) Ground Truth (b) *SGDS* (c) PIS+LP (d) TB+Expl.+LS (e) iDEM

Figure 5: Mode coverage comparison on 40GMM.



(a) Ground Truth (b) AIS



(c) *SGDS* ($d = 32$) (d) *SGDS* ($d = 64$) (e) *SGDS* ($d = 128$)

Figure 6: KDE figures of AIS($T = 100$), ours, and true samples on Manywell-32/64/128.

EUBO, their gap, and the number of energy calls required during training. We employ the same baseline methods and trajectory configurations (including trajectory length and training objectives) as in the high-dimensional settings. We provide detailed configurations, including diffusion scales for each task, in Appendix A.

Results. As demonstrated in Table 6, our method achieves competitive performance on lower-dimensional standard tasks, producing EUBO and ELBO metrics comparable to the strongest baselines, while using significantly fewer energy calls. On the 40GMM task, despite some baselines reporting strong ELBO and EUBO scores, they notably fail to capture the mode located at the bottom-right corner (see Figure 5). In contrast, our framework reliably identifies all modes without sacrificing performance metrics. We report both the first-round and second-round performances of our method, showing that our method attains robust performance on low-dimensional tasks even in the first round, with a slight but consistent improvement observed in the second round.

B.2 Debiasing of Learner from MCMC Searcher

To address potential biases inherent in MCMC sampling due to finite-length chains, our framework incorporates both off-policy TB training using samples from the Searcher and on-policy TB training. This design choice aims to mitigate biases arising from the Searcher samples alone by enabling the Learner model to adjust toward the target distribution.

To evaluate whether the Learner effectively debiases the samples collected by the Searcher, we compare kernel density estimations (KDE) of samples obtained by the AIS Searcher with those

generated by the on/off-policy TB Learner on Manywell distributions. Figure 6 illustrates these KDE comparisons across dimensions 32, 64, and 128.

Due to the varying mode masses assigned in Manywell distributions, even when AIS successfully covers all modes with limited budgets, it struggles to precisely capture the relative mode masses. In contrast, the KDE of the samples generated by the Learner aligns more closely with the true density, effectively reflecting the relative importance of different modes. This result highlights the effectiveness of combining on- and off-policy training to achieve better density approximation than relying solely on finite-budget AIS samples.

C Limitations

While our framework demonstrates strong empirical performance, several limitations remain.

First, the effectiveness of intrinsic rewards from RND depends on careful tuning of the novelty scale parameter α. Poorly calibrated α can overly emphasize exploration, producing noisy or irrelevant samples, or conversely yield overly conservative exploration. This could be mitigated by employing adaptive strategies that dynamically adjust α during sampling based on diversity metrics or exploration progress signals.

Additionally, the quality of samples provided by the Searcher sets a fundamental exploration limit. If the Searcher fails to adequately explore challenging modes, the Learner will inevitably inherit these limitations, particularly in high-barrier energy landscapes. Introducing enhanced exploration strategies, such as parallel tempering or more advanced proposal schemes like HMC, could improve coverage of hard-to-sample modes.