Particle Denoising Diffusion Sampler

Angus Phillips [1] **Hai-Dang Dau** [1] **Michael John Hutchinson** [1] **Valentin De Bortoli** [2] **George Deligiannidis** [1] **Arnaud Doucet** [1]

Abstract

Denoising diffusion models have become ubiquitous for generative modeling. The core idea is to transport the data distribution to a Gaussian by using a diffusion. Approximate samples from the data distribution are then obtained by estimating the time-reversal of this diffusion using score matching ideas. We follow here a similar strategy to sample from unnormalized probability densities and compute their normalizing constants. However, the time-reversed diffusion is here simulated by using an original iterative particle scheme relying on a novel score matching loss. Contrary to standard denoising diffusion models, the resulting Particle Denoising Diffusion Sampler (PDDS) provides asymptotically consistent estimates under mild assumptions. We demonstrate PDDS on multimodal and high dimensional sampling tasks.

1. Introduction

Consider a target probability density π on $\mathbb{R}^d$ of the form

$$\pi(x) = \frac{\gamma(x)}{\mathcal{Z}}, \qquad \mathcal{Z} = \int_{\mathbb{R}^d} \gamma(x)\mathrm{d}x, \tag{1}$$

where $\gamma : \mathbb{R}^d \to \mathbb{R}^+$ can be evaluated pointwise but its normalizing constant $\mathcal{Z}$ is intractable. We develop here a Monte Carlo scheme to sample approximately from π and estimate $\mathcal{Z}$.

We follow an approach inspired by denoising diffusion models (Ho et al., 2020; Song et al., 2021; Song & Ermon, 2019) by considering a "noising" diffusion progressively transporting the original target to a Gaussian. The time-reversal of this diffusion, the "denoising" diffusion, allows us theoretically to sample from the target starting from Gaussian noise. However, it is impossible to simulate this process exactly as its drift depends on the gradient of the logarithm of the intractable marginal densities of the noising diffusion, i.e. the score. For generative modeling, where one has access to samples from π, one can rely on neural networks and score matching (Hyvärinen, 2005; Vincent, 2011). This strategy is not applicable in the Monte Carlo sampling context as we cannot sample the "noising" diffusion since we do not have access to any samples from π to approximate the initial distribution of the diffusion with.

The idea of using denoising diffusion models for Monte Carlo sampling has already been explored by Berner et al. (2022); McDonald & Barron (2022); Vargas et al. (2023); Huang et al. (2024); Richter et al. (2024); Zhang et al. (2024). Berner et al. (2022); Richter et al. (2024); Vargas et al. (2023) focus on the minimization of a reverse Kullback–Leibler divergence or log-variance criterion while McDonald & Barron (2022) rely on an importance sampling scheme which scales poorly in high dimensions. Finally, Huang et al. (2024) relies on a series of Markov chain Monte Carlo (MCMC) to estimate the score. This last scheme does not provide estimates of normalizing constants.

We develop here an alternative approach inspired by denoising diffusion models with guidance. Guided diffusions combine pre-trained diffusion models with a guidance term derived from a likelihood to sample approximately from posterior distributions; see e.g. Song et al. (2021); Chung et al. (2023); Song et al. (2023); Corso et al. (2023). While they provide samples with appealing perceptual properties, they rely on various approximations, in order of importance: (1) approximation of the score and guidance terms, (2) time-discretization of the diffusion and (3) approximate initialization of the diffusion. Wu et al. (2023); Cardoso et al. (2024) have used particle methods also known as Sequential Monte Carlo (SMC) (Doucet et al., 2001; Chopin & Papaspiliopoulos, 2020) to obtain consistent estimates in the generative modeling context.

Our contributions are as follows:
(1) we adapt guided diffusions to sampling problems,
(2) we provide theoretical results quantifying the error introduced by current guided diffusions in a simple scenario,
(3) we develop an SMC scheme to provide consistent estimates in this setup and establish limit theorems,
(4) we introduce an algorithm that reduces the variance of

[1] University of Oxford [2] CNRS, ENS Ulm. Correspondence to: Angus Phillips <angus.phillips@stats.ox.ac.uk>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

the SMC estimates based on a novel score matching loss. All proofs are postponed to the Appendix.

2. Denoising Diffusions with Guidance

2.1. Noising and denoising diffusions

Consider the following noising diffusion $(X_t)_{t\in[0,T]}$,

$$\mathrm{d}X_t = -\beta_t X_t \mathrm{d}t + \sqrt{2\beta_t}\mathrm{d}W_t, \quad X_0 \sim \pi, \tag{2}$$

where $(W_t)_{t\in[0,T]}$ is a d-dimensional Brownian motion and $\beta_t > 0$. The transition density of this diffusion is given by $p(x_t|x_0) = \mathcal{N}(x_t; \sqrt{1-\lambda_t}x_0, \lambda_t \mathrm{I})$ for $\lambda_t = 1 - \exp\left[-2\int_0^t \beta_s \mathrm{d}s\right]$. We denote by π_t the density of X_t under (2). In practice, we consider $\int_0^T \beta_s \mathrm{d}s \gg 1$, and therefore $\pi_T(x) \approx \mathcal{N}(x; 0, \mathrm{I})$. The diffusion (2) thus transforms $\pi_0 = \pi$ into approximately $\mathcal{N}(0, \mathrm{I})$. If instead we initialize (2) using $p_0(x) = \mathcal{N}(x; 0, \mathrm{I})$, its marginals $(p_t)_{t\in[0,T]}$ satisfy $p_t = p_0$.

The time-reversal $(Y_t)_{t\in[0,T]} = (X_{T-t})_{t\in[0,T]}$ of (2), the denoising diffusion, satisfies $Y_T \sim \pi$ and

$$\mathrm{d}Y_t = [\beta_{T-t}Y_t + 2\beta_{T-t}\nabla \log \pi_{T-t}(Y_t)]\,\mathrm{d}t + \sqrt{2\beta_{T-t}}\mathrm{d}B_t, \tag{3}$$

where $(B_t)_{t\in[0,T]}$ is another Brownian motion; see e.g. Haussmann & Pardoux (1986); Cattiaux et al. (2023). The main idea of denoising diffusions is to sample from π by sampling (3) as $Y_T \sim \pi$ (Ho et al., 2020; Song et al., 2021). However, we cannot simulate (3) exactly as, in order of importance, (1) the score terms $(\nabla \log \pi_t)_{t\in[0,T]}$ are intractable, (2) it is necessary to time-discretize the diffusion and (3) π_T cannot be sampled. We can always use numerical integrators and approximate π_T with a unit Gaussian distribution to mitigate (2) and (3). In generative modeling (1) is addressed by leveraging tools from the score matching literature (Vincent, 2011; Hyvärinen, 2005) and using neural network estimators. In our *sampling* setting we do not have access to access to samples from π but only to its unnormalized density. Therefore, alternative approximations must be developed.

2.2. Denoising diffusions with guidance

For generative modeling, the use of denoising diffusions with guidance terms to sample approximately from posterior distributions has become prominent in the inverse problem literature, see e.g. Song et al. (2021); Chung et al. (2023); Song et al. (2023); Corso et al. (2023). We present here a simple extension of this idea applicable to any target $\pi(x)$ defined by (1) by the rewriting

$$\pi(x) = \frac{p_0(x)g_0(x)}{\mathcal{Z}}, \quad \text{for } \; p_0(x) = \mathcal{N}(x; 0, \mathrm{I}) \tag{4}$$

where $g_0(x_0) = \gamma(x_0)/p_0(x_0)$.

Lemma 2.1. *The following identities hold*

$$\pi_t(x_t) = \frac{p_0(x_t)g_t(x_t)}{\mathcal{Z}}; \nabla \log \pi_t(x_t) = -x_t + \nabla \log g_t(x_t), \tag{5}$$

where

$$g_t(x_t) = \textstyle\int g_0(x_0)p(x_0|x_t)\mathrm{d}x_0, \tag{6}$$

and $p(x_0|x_t) = \mathcal{N}(x_0; \sqrt{1-\lambda_t}x_t, \lambda_t \mathrm{I})$ *is the conditional density of* X_0 *given* $X_t = x_t$ *for the diffusion* (2) *initialized using* $X_0 \sim p_0$.

From Lemma 2.1, it follows that (3) can be rewritten as

$$\begin{aligned}\mathrm{d}Y_t = &[-\beta_{T-t}Y_t + 2\beta_{T-t}\nabla \log g_{T-t}(Y_t)]\,\mathrm{d}t \\ &+ \sqrt{2\beta_{T-t}}\mathrm{d}B_t.\end{aligned} \tag{7}$$

This is akin to having a diffusion model with tractable scores $\nabla \log p_t(x_t) = \nabla \log p_0(x_t) = -x_t$ and with $g_t(x_t)$ as a guidance term.

We stress that this guidance formulation is simply a restatement of Section 2.1 using some new notation. While it is possible to write all the sequel in terms of π_t alone, introducing g_t will make the exposition more intuitive.

2.3. Guidance approximation

The most important source of error when approximating (7) is the lack of a closed form expression for g_t as it involves an intractable integral. In the context of inverse problems, a simple approximation used by (Chung et al., 2023; Song et al., 2023) is given by

$$\begin{aligned}g_t(x_t) \approx g_0(\textstyle\int x_0 p(x_0|x_t)\mathrm{d}x_0) &= g_0(\sqrt{1-\lambda_t}x_t) \\ &:= \hat{g}_t(x_t).\end{aligned} \tag{8}$$

This approximation is good when t is close to 0 or T but crude otherwise, as established by the following result.

Proposition 2.2. *Let* $\pi(x) = \mathcal{N}(x; \mu, \sigma^2)$ *and* $(\beta_t)_{t\in[0,T]}$ *be any schedule satisfying* $\lim_{T\to\infty} \int_0^T \beta_s \mathrm{d}s = \infty$. *Consider the following approximation of* (7)

$$\begin{aligned}\mathrm{d}Z_t^{(T)} = &[-\beta_{T-t}Z_t^{(T)} + 2\beta_{T-t}\nabla \log \hat{g}_{T-t}(Z_t^{(T)})]\mathrm{d}t \\ &+ \sqrt{2\beta_{T-t}}\mathrm{d}B_t, \quad Z_0^{(T)} \sim \mathcal{N}(0, 1).\end{aligned} \tag{9}$$

Then $\lim_{T\to\infty} \mathbb{E}[Z_T^{(T)}] = \mu$ *and* $\lim_{T\to\infty} \mathrm{Var}(Z_T^{(T)}) = 1$ *for* $\sigma = 1$ *and otherwise*

$$\lim_{T\to\infty} \mathbb{E}[Z_T^{(T)}] = \frac{\mu}{1-\sigma^2}(1 - e^{-(1/\sigma^2 - 1)}), \tag{10}$$

$$\lim_{T\to\infty} \mathrm{Var}(Z_T^{(T)}) = \frac{1 - e^{-2(1/\sigma^2-1)}}{2(1/\sigma^2 - 1)}. \tag{11}$$

Hence, even without considering any time discretization error, $\lim_{T\to\infty} \mathbb{E}[Z_T^{(T)}] \neq \mu$ and $\lim_{T\to\infty} \mathrm{Var}(Z_T^{(T)}) \neq \sigma^2$

for $\sigma \neq 1$. If we consider the target $\mathcal{N}(\mu, \sigma^2)^{\otimes d}$, a similar result holds along each dimension. Therefore, the Kullback–Leibler divergence between the target and the result of running (9) grows linearly with d. As a result an exponentially increasing number of samples is needed to obtain an importance sampling approximation of the target of reasonable relative variance (Chatterjee & Diaconis, 2018).

3. Particle Denoising Diffusion Sampler

In this section, we propose a particle method to correct the discrepancy between the distribution outputted by the guided diffusion and the target. Let $[P] = \{1, ..., P\}$ for $P \in \mathbb{N}$. We first present the exact joint distribution of $(X_{t_k})_{k\in\{0,...,K\}}$ with $t_k = k\delta$ for a fixed step size δ for the diffusion (2) where $K = T/\delta$ is an integer. We then make explicit the corresponding reverse-time Markov transitions for this joint distribution and show how they can be approximated. Finally we show how these approximations can be corrected to obtain consistent estimates using SMC. Instead of writing X_{t_k}, we will write X_k to simplify notation. Similarly we will write g_k for g_{t_k}, λ_k for λ_{t_k} etc.

3.1. From continuous time to discrete time

For $k \in [K]$, let $\alpha_k = 1 - \exp\left[-2\int_{(k-1)\delta}^{k\delta} \beta_s \mathrm{d}s\right]$. The joint distribution of $X_{0:K} = (X_0, X_1, ..., X_K)$ under (2) satisfies

$$\pi(x_{0:K}) = \pi(x_0) \textstyle\prod_{k\in[K]} p(x_k|x_{k-1}), \tag{12}$$

for $p(x_k|x_{k-1}) = \mathcal{N}(x_k; \sqrt{1-\alpha_k}x_{k-1}, \alpha_k \mathrm{I})$. This implies in particular

$$\pi(x_k, x_{k+1}) = \pi_k(x_k)p(x_{k+1}|x_k). \tag{13}$$

We also denote by $p(x_{0:K})$ the joint distribution of (2) initialized at $p_0(x_0)$, in this case the Markov process is stationary, i.e. $p_k(x_k) = p_0(x_k)$, and reversible w.r.t. p_0. From Bayes' theorem and equations (5) and (13), the backward transitions of (12) satisfy

$$\begin{aligned}\pi(x_k|x_{k+1}) &= \frac{\pi_k(x_k)p(x_{k+1}|x_k)}{\pi_{k+1}(x_{k+1})} \\ &= \frac{p_0(x_k)p(x_{k+1}|x_k)}{p_0(x_{k+1})}\frac{g_k(x_k)}{g_{k+1}(x_{k+1})} = \frac{p(x_k|x_{k+1})g_k(x_k)}{g_{k+1}(x_{k+1})} \\ &\approx p(x_k|x_{k+1})\exp[\langle\nabla \log g_{k+1}(x_{k+1}), x_k - x_{k+1}\rangle] \\ &= \mathcal{N}(x_k; \sqrt{1-\alpha_{k+1}}x_{k+1} \\ &\qquad + \alpha_{k+1}\nabla \log g_{k+1}(x_{k+1}), \alpha_{k+1}\mathrm{I}),\end{aligned} \tag{14}$$

where we used $g_k \approx g_{k+1}$ and a Taylor expansion of $\log g_{k+1}(x_k)$ around x_{k+1}; both of which are reasonable when $\delta \ll 1$. Since $\alpha_{k+1} \approx 2\beta_{k+1}\delta$ and $\sqrt{1-\alpha_{k+1}} \approx 1 - \beta_{k+1}\delta$, this approximation of $\pi(x_k|x_{k+1})$ corresponds to a discretization of the time-reversal (7).

While this discrete-time representation is interesting, it is clearly typically impossible to exploit it to obtain exact samples from π since, like its continuous time counterpart (7), it relies on the intractable potentials $(g_k)_{k=1}^K$. In the following Section 3.2, given an approximation $\hat{g}_k$ of g_k, we propose a particle mechanism to sample exactly from π as the number of particles goes to infinity.

3.2. From discrete time to particles

We use a particle method to sample from π. The key idea is to break the difficult problem of sampling from π into a sequence of simpler intermediate sampling problems. Ideally we would sample from π_K first then $\pi_{K-1}, \pi_{K-2}, ...$ until $\pi_0 = \pi$. Unfortunately this is not possible as this requires knowing g_k. Suppose that we have an approximation $\hat{g}_k$ of g_k (for instance, the guidance approximation defined in (8)). Inspired by (5), we sample instead for $k \in [K]$ from the sequence of densities

$$\hat{\pi}_k(x_k) \propto p_0(x_k)\hat{g}_k(x_k), \quad \hat{\mathcal{Z}}_k = \textstyle\int p_0(x_k)\hat{g}_k(x_k)\mathrm{d}x_k, \tag{15}$$

backward in time. At $k = K$, the function g_K is almost constant so we choose $\hat{g}_K \equiv 1$ and sampling from $\hat{\pi}_K = \mathcal{N}(0, \mathrm{I})$ is easy. Given particles approximately distributed according to $\hat{\pi}_{k+1}$, we aim to obtain samples from $\hat{\pi}_k$. Drawing inspiration from (14), we sample according to the proposal

$$\begin{aligned}\hat{\pi}(x_k|x_{k+1}) := \mathcal{N}(x_k; &\sqrt{1-\alpha_{k+1}}x_{k+1} \\ &+ \alpha_{k+1}\nabla \log \hat{g}_{k+1}(x_{k+1}), \alpha_{k+1}\mathrm{I}).\end{aligned} \tag{16}$$

This is not the only option: we can also use the exponential integrator

$$\begin{aligned}\hat{\pi}(x_k|x_{k+1}) := \mathcal{N}(x_k; &\sqrt{1-\alpha_{k+1}}x_{k+1} \\ &+ 2(1-\sqrt{1-\alpha_{k+1}})\nabla \log \hat{g}_{k+1}(x_{k+1}), \alpha_{k+1}\mathrm{I}).\end{aligned}$$

We could use instead the Euler integrator for (7) but it is clear that the latter would induce greater error. We then reweight the pairs (x_k, x_{k+1}) using the weights

$$\begin{aligned}w_k(x_k, x_{k+1}) &:= \frac{\hat{\pi}_k(x_k)p(x_{k+1}|x_k)}{\hat{\pi}_{k+1}(x_{k+1})\hat{\pi}(x_k|x_{k+1})} \\ &\propto \frac{\hat{g}_k(x_k)}{\hat{g}_{k+1}(x_{k+1})\hat{\pi}(x_k|x_{k+1})}\frac{p_0(x_k)p(x_{k+1}|x_k)}{p_0(x_{k+1})} \\ &= \frac{\hat{g}_k(x_k)p(x_k|x_{k+1})}{\hat{g}_{k+1}(x_{k+1})\hat{\pi}(x_k|x_{k+1})},\end{aligned} \tag{17}$$

where the second line follows from (15). The first line of (17) means that the x_k marginal of the weighted system consistently approximates $\hat{\pi}_k$. It should be noted that w_k quantifies the error in (16).

Finally, we resample these particles with weights proportional to (17). This resampling operation allows us to focus

Algorithm 1 Particle Denoising Diffusion Sampler

```
Input: Schedule $(\beta_t)_{t\in[0,T]}$ as in (2); Approximations $(\hat{g}_k)_{k=0}^{K}$ s.t. $\hat{g}_0 = g_0, \hat{g}_K = 1$; Number of particles $N$
  Sample $X_K^i \overset{\text{iid}}{\sim} \mathcal{N}(0, \mathrm{I})$ for $i \in [N]$
  Set $\hat{\mathcal{Z}}_K \leftarrow 1$ and $\omega_K^i \leftarrow 1/N$ for $i \in [N]$
  for $k = K-1, \ldots, 0$ do
    Move. Sample $\tilde{X}_k^i \sim \hat{\pi}(\cdot|X_{k+1}^i)$ for $i \in [N]$ (see (16))
    Weight. $\omega_k^i \leftarrow \omega_k(\tilde{X}_k^i, X_{k+1}^i)$ for $i \in [N]$ (see (17))
    Set $\hat{\mathcal{Z}}_k \leftarrow \hat{\mathcal{Z}}_{k+1} \times \frac{1}{N}\sum_{i\in[N]}\omega_k^i$
    Normalize $\omega_k^i \leftarrow \omega_k^i / \sum_{j\in[N]}\omega_k^j$
    Resample. $X_k^{1:N} \leftarrow \mathrm{resample}(\tilde{X}_k^{1:N}, \omega_k^{1:N})$ (see Section 3.3)
    MCMC (Optional). Sample $X_k^i \leftarrow \mathfrak{M}_k(X_k^i, \cdot)$ for $i \in [N]$ using a $\hat{\pi}_k$-invariant MCMC kernel $\mathfrak{M}_k$.
  end for
Output: Estimates $\hat{\pi}^N = \frac{1}{N}\sum_{i\in[N]}\delta_{X_0^i}$ of $\pi$, $\hat{\mathcal{Z}}_0^N$ of $\mathcal{Z}$
```

the computational efforts on promising regions of the space but some particles are replicated multiple times, reducing the population diversity. Therefore, we then optionally perturb the resampled particles using a MCMC kernel of invariant distribution $\hat{\pi}_k$. The resulting Particle denoising diffusion sampler (PDDS) is summarized in Algorithm 1.

3.3. Algorithm settings

Reparameterization. In practice, we would like to have p_0 to be such that g_0 is bounded or has bounded moments. To achieve this, it can be desirable to obtain a variational approximation $\mathcal{N}(x; \mu, \Sigma)$ of π then do the change of variables $x' = \Sigma^{-1/2}(x - \mu)$, sample in this space using PDDS before mapping the samples back using $x = \mu + \Sigma^{1/2}x'$.

Resampling. The idea of resampling is to only propagate particles in promising regions of the state space. Given N particles $\tilde{X}_k^{1:N}$ and N weights $\omega_k^{1:N}$ summing to 1, resampling selects N output particles $X_k^{1:N}$ such that, for any function $\varphi : \mathbb{R}^d \to \mathbb{R}$,

$$\mathbb{E}\Big[\tfrac{1}{N}\textstyle\sum_{i\in[N]}\varphi(X_k^i)|\tilde{X}_k^{1:N}, \omega_k^{1:N}\Big] = \textstyle\sum_{i\in[N]}\omega_k^i\varphi(\tilde{X}_k^i).$$

Popular schemes satisfying this identity are multinomial, stratified, residual, and systematic resampling (Douc & Cappé, 2005). We employ systematic resampling in all our simulations as it provides the lowest variance estimates.

Resampling can however reduce particle diversity by introducing identical particles in the output. As such, a popular recipe is to trigger resampling at time k only when the Effective Sample Size (ESS), a measure of particle diversity defined by $(\sum_{i\in[N]}(\omega_k^i)^2)^{-1}$, is below a certain threshold (Del Moral et al., 2012; Dai et al., 2022). This is implemented using Algorithm 3 presented in Appendix A.

MCMC kernel. We want to design an MCMC kernel of invariant distribution $\hat{\pi}_k(x_k)$ defined in (15). As we have access to $\nabla\log\hat{\pi}_k(x_k) = -x_k + \nabla\log\hat{g}_k(x_k)$, we can use a Metropolis-adjusted Langevin algorithm (MALA) or Hamiltonian Monte Carlo; e.g. MALA considers a proposal

$$x_k^\star = x_k + \gamma\nabla\log\hat{\pi}_k(x_k) + \sqrt{2\gamma}\epsilon, \quad \epsilon \sim \mathcal{N}(0, \mathrm{I}),$$

for a step size γ. This proposal is accepted with probability

$$\min\left\{1, \frac{\hat{\pi}_k(x_k^\star)\mathcal{N}(x_k; x_k^\star + \gamma\nabla\log\hat{\pi}_k(x_k^\star), 2\gamma\mathrm{I})}{\hat{\pi}_k(x_k)\mathcal{N}(x_k^\star; x_k + \gamma\nabla\log\hat{\pi}_k(x_k), 2\gamma\mathrm{I})}\right\}.$$

3.4. Theoretical results

Fixed number of discretization steps. We show below that the estimates $\hat{\mathcal{Z}}_0^N$ and $\pi^N f = \frac{1}{N}\sum_{i=1}^N f(X_0^i)$ of Algorithm 1 satisfy a central limit theorem. This follows from standard SMC theory (Del Moral, 2004; Webber, 2019).

Proposition 3.1. *Assume that $\mathbb{E}[w_k(X_k, X_{k+1})^2] < \infty$ where the expectation is w.r.t. $\hat{\pi}(x_{k+1})\hat{\pi}(x_k|x_{k+1})$ and that $\int \pi_k(x)(g_k/\hat{g}_k)(x)\mathrm{d}x < \infty$. Then $\hat{\mathcal{Z}}_0^N$ is an unbiased estimate of $\mathcal{Z}$ and has finite variance. If multinomial resampling is used at every step, $\sqrt{N}(\hat{\mathcal{Z}}_0^N/\mathcal{Z} - 1)$ is asymptotically normal with asymptotic variance*

$$\sigma_K^2 = \chi^2(\pi_K||\hat{\pi}_K) + \sum_{k=0}^{K-1}\chi^2(\pi_k(x_k)\pi(x_k|x_{k+1})||\hat{\pi}_k(x_k)\hat{\pi}(x_k|x_{k+1})),$$

with $\chi^2(\cdot||\cdot)$ the chi-squared divergence between two distributions. Moreover, for any bounded function f, we also have asymptotic normality of $\sqrt{N}(\hat{\pi}^N f - \pi f)$.

The finiteness assumptions on $\mathbb{E}[w_k(X_k, X_{k+1})^2]$ and $\int \pi_k(x)(g_k/\hat{g}_k)(x)\mathrm{d}x$ require that $\hat{g}_k$ is not too far from g_k but still allow enough freedom in the choice of $\hat{g}_k$. The expression for σ_K^2 suggests that choosing $\hat{g}_k$ close to g_k will reduce the asymptotic variance as (16) will better approximate (14).

Infinitely fine discretization limit. We now investigate the performance of PDDS as the number of discretization time steps goes to infinity. Even when all the weights are equal, multinomial resampling still kills over a third of particles on average (Chopin & Papaspiliopoulos, 2020). Hence a fine discretization with repeated applications of multinomial resampling leads to the total collapse of the particle approximation. This has been formalized in the continuous-time setting (Chopin et al., 2022). In our case, it can be readily checked that $\sigma_K^2 \to \infty$ as $K \to \infty$ in general. This justifies using a more sophisticated resampling strategy. Indeed, when using the sorted stratified resampling strategy of Gerber et al. (2019), the following results show that our particle approximations remain well behaved as $K \to \infty$.

Proposition 3.2. *Consider the setting of Proposition 3.1 with sorted stratified resampling. Then there exists a sequence of sets (B_N) such that $\mathbb{P}(B_N) \to 1$ and*

$$\limsup_{N\to\infty} \mathbb{E}[N(\hat{\mathcal{Z}}_0^N/\mathcal{Z}-1)^2 \mathbb{1}_{B_N}] \leq \zeta_K^2$$

with

$$\zeta_K^2 := \chi^2(\pi_K||\hat{\pi}_K)+$$
$$\sum_{k=0}^{K-1} \int \frac{\pi_{k+1}(x_{k+1})^2}{\hat{\pi}_{k+1}(x_{k+1})} \chi^2(\pi(x_k|x_{k+1})||\hat{\pi}(x_k|x_{k+1}))\mathrm{d}x_{k+1}.$$

The next result bounds the limit when $K \to \infty$, i.e. $\delta = T/K \to 0$

Proposition 3.3. *Under the setting of Proposition 3.2, assume that $\beta_t \equiv 1$ and that the target distribution satisfies the regularity conditions in Appendix C.4.1. Let $\bar{g}_t := g_t/\mathcal{Z}$ and $\tilde{g}_t := \hat{g}_t/\int p_0(x)\hat{g}_t(x)\mathrm{d}x$. Assume further that the approximations $\hat{g}_t, \bar{g}_t, \tilde{g}_t$ satisfy $C_1^{-1} \leq \bar{g}_t/\tilde{g}_t \leq C_1$ and $\|\nabla \log \hat{g}_t(x_t)\| \leq C_2(1+\|x_t\|)$ for some $C_1, C_2 \geq 0$. Then*

$$\limsup_{K\to\infty} \zeta_K^2 \leq \chi^2(\pi_T||\hat{\pi}_T)+$$
$$2\int_0^T \mathbb{E}_{X_t\sim\pi_t}\left[\frac{\bar{g}_t}{\tilde{g}_t}(X_t)\|\nabla \log g_t(X_t) - \nabla \log \hat{g}_t(X_t)\|^2\right]\mathrm{d}t.$$

This result highlights precisely how the asymptotic error depends on the quality of the estimates of g_t and its logarithmic gradient. Let $\pi^{\hat{g}}(\mathrm{d}x_{[0,T]})$ be the path measure induced by running (7) with some approximation $\hat{g}_t$, i.e. the distribution of $(Y_t)_{t\in[0,T]}$ given by (7). If importance sampling were directly used to correct between $\pi^{\hat{g}}(\mathrm{d}x_{[0,T]})$ and $\pi(\mathrm{d}x_{[0,T]})$, the asymptotic error for $\hat{\mathcal{Z}}_0^N/\mathcal{Z}$ would be equal to $\chi^2(\pi||\pi^{\hat{g}})$ which is greater than $\exp\{\mathrm{KL}(\pi|\pi^{\hat{g}})\} - 1$. In contrast, ignoring the negligible first term $\chi^2(\pi_T||\hat{\pi}_T)$, the error in this proposition is upper bounded by $2C_1\mathrm{KL}(\pi|\pi^{\hat{g}})$. This shows how PDDS helps reduce the error of naive importance sampling.

4. Learning Potentials via Score Matching

The previous theoretical results establish the consistency of PDDS estimates for any reasonable approximation $\hat{g}_k$ of g_k but also show that better approximations lead to lower Monte Carlo errors. We show here how to use the approximation of π outputted by PDDS to learn a better neural network (NN) approximation $\hat{g}_\theta(k, x_k)$ of the potential functions g_k by leveraging score matching ideas. Once we have learned those approximations, we can then run again PDDS (Algorithm 1) with the new learned potentials.

4.1. Different score identities

We follow the notation outlined at the beginning of Section 3. From (5), we have the identity

$$\nabla \log g_k(x_k) = x_k + \nabla \log \pi_k(x_k). \tag{18}$$

Hence, if we obtain an approximation $\nabla \log \pi_\theta(k, x_k)$ of $\nabla \log \pi_k(x_k)$, then we get an approximation $\log \hat{g}_\theta(k, x_k) = \frac{1}{2}||x_k||^2 + \log \hat{\pi}_\theta(k, x_k)$ of $\log g_k(x_k)$.

To learn the score, we rely on the following result.

Proposition 4.1. *The score satisfies the standard Denoising Score Matching (DSM) identity*

$$\nabla \log \pi_k(x_k) = \textstyle\int \nabla \log p(x_k|x_0)\, \pi(x_0|x_k)\mathrm{d}x_0. \tag{19}$$

Moreover, if $\int \|\nabla \pi(x_0)\| e^{-\eta\|x_0\|^2}\mathrm{d}x_0 < \infty, \forall \eta > 0$, the Novel Score Matching (NSM) identity

$$\nabla \log \pi_k(x_k) = \kappa_k \textstyle\int \nabla \log g_0(x_0)\, \pi(x_0|x_k)\mathrm{d}x_0 - x_k \tag{20}$$

holds for $\kappa_k = \sqrt{1-\lambda_k}$ and $\nabla \log g_0(x_0) = \nabla \log \pi_0(x_0) + x_0$. Hence, we can approximate the score by minimizing one of the two following loss functions

$$\ell_{\text{DSM}}(\theta) = \sum_{k\in[K]} \mathbb{E}\|\nabla \log \hat{\pi}_\theta(k, X_k) - \nabla \log p(X_k|X_0)\|^2,$$
$$\ell_{\text{NSM}}(\theta) = \sum_{k\in[K]} \mathbb{E}||\nabla \log \hat{\pi}_\theta(k, X_k) + X_k$$
$$- \kappa_k \nabla \log g_0(X_0)||^2,$$

where the first loss is applicable if π has finite second moment and the second loss is applicable if additionally $\mathbb{E}_\pi[||\nabla \log \pi(X)\|^2] < \infty$. All the expectations are taken w.r.t. $\pi(x_0)p(x_k|x_0)$. For expressive neural networks, both $\ell_{\text{DSM}}(\theta)$ and $\ell_{\text{NSM}}(\theta)$ are such that $\nabla \log \hat{\pi}_\theta(k, x_k) = \nabla \log \pi_k(x_k)$ at the minimizer.

The benefit of this novel loss is that it is much better behaved compared to the standard denoising score matching loss when $\delta \ll 1$ as established below, this is due to the fact that the variance of the terms appearing in ℓ_{DSM} for k close to zero become very large. This loss is not applicable to generative modeling as π is only available through samples.

4.2. Benefits of alternative score matching identity

We establish here formally the benefits of NSM over DSM. NSM score matching prevents variance blow up as k is close to time 0 and $\delta \to 0$. This is more elegantly formalized in continuous-time. In this case, the renormalized loss functions ℓ_{DSM} and ℓ_{NSM} introduced in Proposition 4.1 become for $s_\theta(t, x_t) = \nabla \log \hat{\pi}_\theta(t, x_t)$

$$\ell_{\text{DSM}}(\theta) = \textstyle\int_0^T \mathbb{E}\|s_\theta(t, X_t) - \nabla \log p(X_t|X_0)\|^2 \mathrm{d}t,$$
$$\ell_{\text{NSM}}(\theta) = \textstyle\int_0^T \mathbb{E}||s_\theta(t, X_t) + X_t - \kappa_t \nabla \log g_0(X_0)||^2 \mathrm{d}t,$$

where the expectations are w.r.t. $\pi(x_0)p(x_t|x_0)$. Unbiased estimates of the gradient of these losses $\hat{\nabla}\ell_{\text{DSM}}(\theta)$ and $\hat{\nabla}\ell_{\text{NSM}}(\theta)$ are obtained by computing the gradient w.r.t. θ of the argument within the expectation at a sample $\tau \sim \text{Unif}[0, T]$, $X_0 \sim \pi$ and $X_\tau \sim p(x_\tau|X_0)$.

The following result clarifies the advantage of NSM score matching.

Proposition 4.2. *Let $d = 1$, $\beta_t \equiv 1$, and $\pi(x_0) = \mathcal{N}(x_0; \mu, \sigma^2)$. Suppose that the score network $s_\theta(t, x_t)$ is a continuously differentiable function jointly on its three variables. Moreover, assume that $|s| + \|\nabla_\theta s\| \leq C(1 + |\theta| + |t| + |x_t|)^\alpha$ for some $C, \alpha > 0$; and that $\mathbb{E}_\pi[||\nabla_\theta s_\theta(0, X_0)||^2] > 0$ for all θ. Then, for all θ, $\ell_{\text{DSM}}(\theta)$ and $\mathbb{E}[||\hat{\nabla}\ell_{\text{DSM}}(\theta)||^2]$ are infinite whereas $\ell_{\text{NSM}}(\theta)$ and $\mathbb{E}[||\hat{\nabla}\ell_{\text{NSM}}(\theta)||^2]$ are finite.*

4.3. Neural network parametrization

Contrary to standard practice for generative modeling (Ho et al., 2020; Song et al., 2021), we parameterize a function and not a vector field. This is necessary to run PDDS (Algorithm 1) as it relies on potentials to weight particles.

We will use a parameterization of the form $\log \hat{\pi}_\theta(k, x_k) = \log \hat{g}_\theta(k, x_k) - \frac{1}{2}||x||^2$ where we parametrize $\hat{g}_\theta$ using two neural networks r_η and N_γ such that $\theta = (\eta, \gamma)$. The network r_η returns a scalar whereas N_γ returns a vector in $\mathbb{R}^d$. The precise expression is given by

$$\begin{aligned}\log \hat{g}_\theta(k, x_k) = {} & [r_\eta(k) - r_\eta(0)] \langle N_\gamma(k, x_k), x_k\rangle + \\ & + [1 - r_\eta(k) + r_\eta(0)] \log g_0(\sqrt{1 - \lambda_k} x_k).\end{aligned}$$

This parametrization takes advantage of the simple approximation (8) with which it coincides at time 0. Zhang & Chen (2022) used a similar parametrization to incorporate gradient information in their control policy. Although $\log \hat{g}_\theta(k, x_k)$ is a scalar, we deliberately let $N_\gamma(k, x_k)$ return a vector which is then scalar-multiplied with x_k. This is usually done in the literature when the scalar potential instead of just its gradient is learned (see e.g. Salimans & Ho, 2021) and helps improve model expressiveness.

4.4. Training the neural network

Both $\ell_{\text{DSM}}(\theta)$ and $\ell_{\text{NSM}}(\theta)$ can be written as $\sum_{k \in [K]} \mathbb{E}\left[\ell(\theta, k, X_0, X_k)\right]$ for appropriately defined local loss functions ℓ. Algorithm 2 describes the gradient updates according to the batch size B.

In practice, we first run Algorithm 1 with a simple approximation such as (8). We then use Algorithm 2 to learn $\hat{g}_\theta(k, x_k)$ and can then execute Algorithm 1 again with $\hat{g}_k(x_k) = \hat{g}_\theta(k, x_k)$ to obtain lower variance estimates of π and $\mathcal{Z}$. We can further refine the approximation of g_k using Algorithm 2. In practice we found that one or two iterations with larger N_{up} are sufficient, although more frequent iterations with smaller N_{up} can give better performance under a limited budget of target density evaluations.

Algorithm 2 Potential neural network training

```
Input: Particle approx. $\hat{\pi}^N$ outputted by Algorithm 1; Potential NN $\hat{g}_k(\theta, x_k)$; Initialization $\theta_0$; Local loss functions $\ell(\theta, k, x_0, x_k)$; Batch size $B$; Number of gradient updates $N_{\text{up}}$; Learning rate $\eta > 0$.
  for $i = 1, 2, \ldots, N_{\text{up}}$ do
    for $b = 1, 2, \ldots, B$ do
      Sample $X_0^b \sim \hat{\pi}^N(\cdot)$; $k_b \sim \text{Unif}[K]$;
      Sample $X_{k_b} \sim p_{k_b,0}(\cdot|X_0^b)$
    end for
    $\theta_i := \theta_{i-1} - \frac{\eta}{B} \sum_{b \in [B]} \nabla_\theta \ell(\theta_{i-1}, k_b, X_0^b, X_{k_b})$
  end for
Output: Potential $\hat{g}_{\theta_{N_{\text{up}}}}(k, x_k)$
```

4.5. Mechanisms behind potential improvement

It could be unclear at first sight how the iterative procedure described in Section 4.4 leads to an improvement of $\hat{g}_k(x_k)$. Superficially, it seems like we try to improve a poor potential approximation using particles produced by the poor approximation itself. However, results in Section 3.4 imply that the SMC mechanism provides a consistent estimate of the target for *any* approximation of the potential. Thus, we expect that for N large enough, the output of Algorithm 1 would have higher quality than the particles used to learn the current $\hat{g}_k$. This mechanism has been studied in a different setting in Heng et al. (2020) and it would be interesting to extend their results to our case.

Moreover, the training process uses not only the output of Algorithm 1, but also further information about the target injected via a variety of mechanisms (the guidance loss described in Section 4.1 and the neural network parametrization described in Section 4.3). Quantifying the gain from these techniques is an open question. We provide ablation studies in Appendix D.3.

5. Related Work

SMC samplers (Del Moral et al., 2006; Dai et al., 2022) are a general methodology to sample sequentially from a sequence of distributions. They rely on the notion of forward and backward kernels in order to move from one distribution to another. PDDS can be cast in this framework where the forward kernel is chosen to be the forward noising diffusion and the backward kernel the approximate time-reversal. The novelty of our work is the exploitation of the special structure induced by such a choice to come up with efficient backward kernel estimates. Other standard methods include Annealed Importance Sampling (Neal, 2001) and Parallel Tempering (Geyer, 1991). Unlike our work, all these methods approximate a sequence of tempered versions of the target. While they are standard, it is also well-known that tempering strategies can exhibit poor performance for mul-

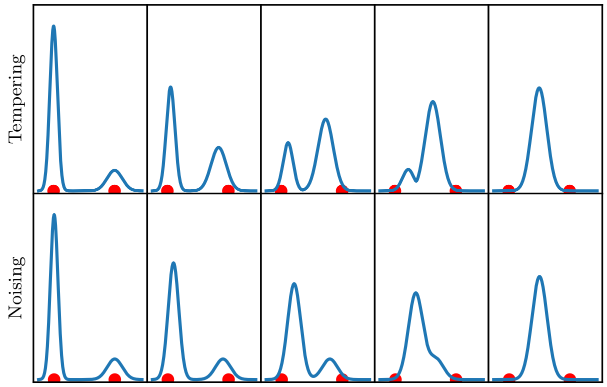

Figure 1: Tempered (top) and noised (bottom) sequences of distributions for the target $\pi(x) = 0.8\mathcal{N}(x; -4, 0.5^2) + 0.2\mathcal{N}(x; 4, 1)$. The tempered sequence follows $\pi_t(x) \propto \pi(x)^{(1-\eta_t)}\phi(x)^{\eta_t}$ where ϕ is the standard normal and η_t increases from 0 to 1. The noising sequence follows the forward diffusion in Equation (2). Red dots indicate the position of modes in the original target. The tempered sequence suffers from mode switching, i.e. the low mass large width mode becomes dominant across the tempered path. The noised sequence does not suffer from this problem.

timodal targets (Woodard et al., 2009; Tawn et al., 2020; Syed et al., 2022) as tempering can change dramatically the masses of distribution modes depending on their widths. Adding noise does not suffer from this issue (Máté & Fleuret, 2023). It only perturbs the distribution locally, preserving the weights of the modes; see Figure 1 for an illustration.

Our sampler can also be interpreted as sampling from a sequence of distributions using so-called twisted proposals. The general framework for approximating these twisted proposals using SMC has been considered in Guarniero et al. (2017); Heng et al. (2020); Lawson et al. (2022). In particular, Wu et al. (2023) and Cardoso et al. (2024) recently apply such ideas to conditional simulation in generative modeling given access to a pretrained score network. Wu et al. (2023) rely on the simple approximation (8) and do not quantify its error, while Cardoso et al. (2024) use a different proposal kernel which leverages the structure of a linear inverse problem. Our setup is here different as we consider general Monte Carlo sampling problems. We do not rely on any pretrained network and refine our potential approximations using an original loss function. We additionally provide theoretical results in particular when the discretization time step goes to zero.

6. Experimental Results

6.1. Normalizing constant estimation

We evaluate the quality of normalizing constant estimates produced by PDDS on a variety of sampling tasks. We

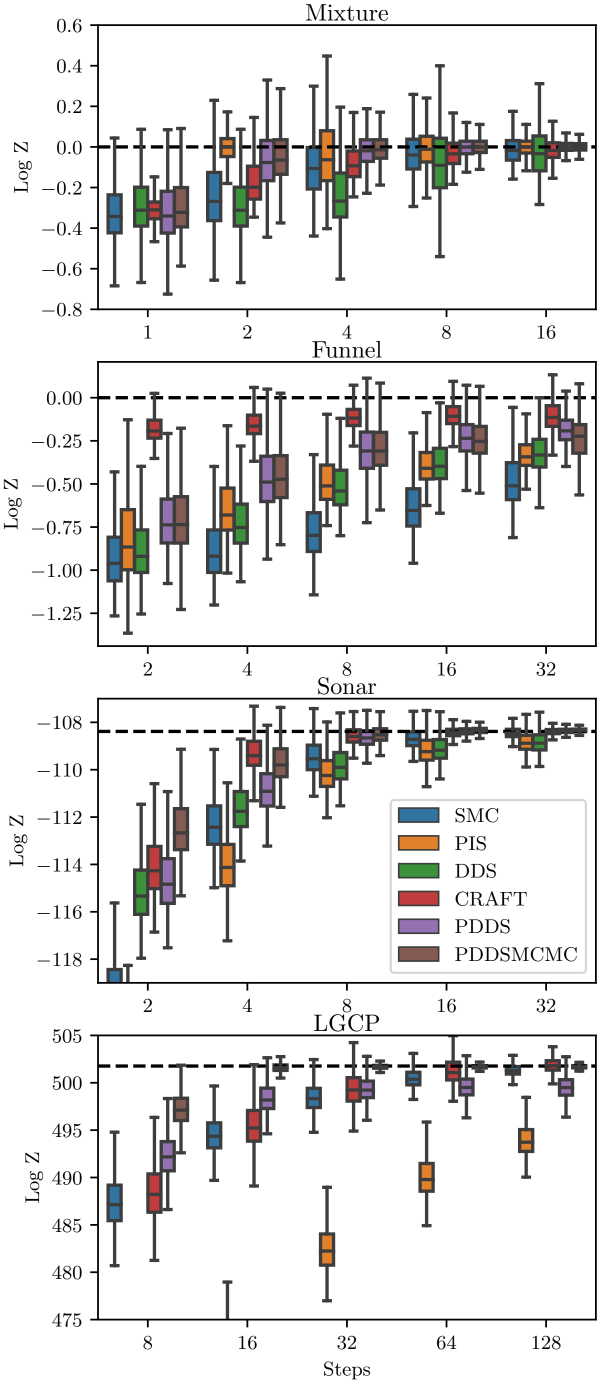

Figure 2: $\log \hat{\mathcal{Z}}_0^N$ for our method (PDDS and PDDSMCMC), compared with SMC, CRAFT, DDS and PIS. Dotted black represents analytic ground truth where available, otherwise long-run SMC. Variation is displayed over both training and sampling seeds (2000 total). The y-axes on Sonar and LGCP have been cropped and outliers (present in all methods) removed for clarity. Uncurated samples are presented in Appendix D.4.

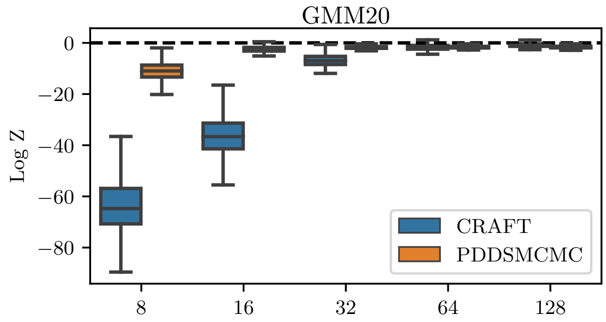

Figure 3: $\log \hat{\mathcal{Z}}_0^N$ for CRAFT and PDDS-MCMC on the GMM task in 20 dimensions. Variation displayed over training and sampling seeds (1000 total).

consider two synthetic target densities and two posterior distributions, with results on additional targets included in Appendix D.4. The synthetic targets are Mixture, a 2-dimensional mixture of Gaussian distributions with separated modes (Arbel et al., 2021) and Funnel, a 10-dimensional target displaying challenging variance structure (Neal, 2003). The posterior distributions are Sonar, a logistic regression posterior fitted to the Sonar (61-dimensional) dataset and LGCP, a log Gaussian Cox process (Møller et al., 1998) modelling the rate parameter of a Poisson point process on a $40 \times 40 = 1600$-point grid, fitted to the Pines dataset. Precise specification of these targets can be found in Appendix D.1.

We compare our performance to a selection of strong baselines. We consider two annealing algorithms, firstly an SMC sampler (Del Moral et al., 2006) with HMC kernels and secondly CRAFT (Matthews et al., 2022), which uses normalizing flows to transport particles at each step of an SMC algorithm. We also consider two diffusion-based sampling methods, Path Integral Sampler (PIS) (Zhang & Chen, 2022) and Denoising Diffusion Sampler (DDS) (Vargas et al., 2023). As mentioned in Section 3.3, we reparameterize the target using a variational approximation for all methods. We include hyperparameter and optimizer settings, run times, and experimental procedures in Appendix D.2.

We present the normalizing constant estimation results in Figure 2. PDDS uses the same training budget as PIS and DDS. We also include PDDS with optional MCMC steps (PDDS-MCMC). Considering the posterior sampling tasks, PDDS-MCMC is the best performing method in terms of estimation bias and variance everywhere, except for 4 steps on the Sonar task where CRAFT performs the best. PDDS without MCMC steps performs on par with CRAFT on average, specifically PDDS outperforms CRAFT for larger step regimes on Sonar and low step regimes on LGCP while CRAFT outperforms PDDS in the opposite regimes. Both PDDS methods uniformly outperform the diffusion-based approaches (PIS and DDS). Considering the synthetic target densities, CRAFT is the best performing method on the synthetic Funnel task while PDDS and PDDS-MCMC are the best performing methods on Mixture. Our approach again outperforms both of the diffusion-based approaches.

While CRAFT performs competitively with our method on certain tasks, we note that CRAFT cannot be easily refined. Indeed, if we want more intermediate distributions between the reference and the target, we can simply decrease the discretization step size for PDDS, but would need to relearn the flows for CRAFT. In addition, choosing the flow structure in CRAFT can be challenging and is problem-specific. In contrast, we used the same simple MLP structure for all tasks for PDDS. Finally, CRAFT relies on MCMC moves to inject noise into the system and prevent particle degeneracy. On the other hand PDDS can produce competitive results without MCMC, with the option of boosting performance with MCMC steps if the computational budget allows.

We further compared PDDS-MCMC with CRAFT on a challenging Gaussian Mixture Model (GMM) with 40 highly separated modes (Midgley et al., 2023) in a range of dimensions. We present the normalising constant estimates for the task in 20 dimensions in Figure 3. We observe that PDDS-MCMC significantly outperforms CRAFT in bias and variance, particularly in low step regimes. Furthermore, PIS and DDS failed to produce competitive results. Results in additional dimensions are included in Appendix D.4.

6.2. Sample quality

We also visually assess the quality of samples from each method in Figure 4. We choose the multi-modal Mixture task in 2-dimensions for ease of visualization. Unsurprisingly we find that both PIS and DDS do not capture all 6 modes of the distribution due to the mode-seeking behaviour of the reverse KL objective. Samples from our approach appear of similar quality to those of SMC and CRAFT.

We further quantitatively compared the sample quality of our method versus CRAFT by evaluating the entropy-regularized Wasserstein-2 distance between samples from the model and the target in the challenging GMM task. We present the results in Figure 6 for the task in 20 dimensions. Here PDDS-MCMC clearly outperforms CRAFT producing samples with significantly lower transport cost to the target distribution. From the sample visualization we observe that PDDS-MCMC is able to correctly recover a far greater proportion of the target modes than our CRAFT implementation.

6.3. Iterations of potential approximation

Here we demonstrate the improvement in normalizing constant estimates due to our iterative potential approximation scheme. Figure 5 shows the improvement in ESS and re-

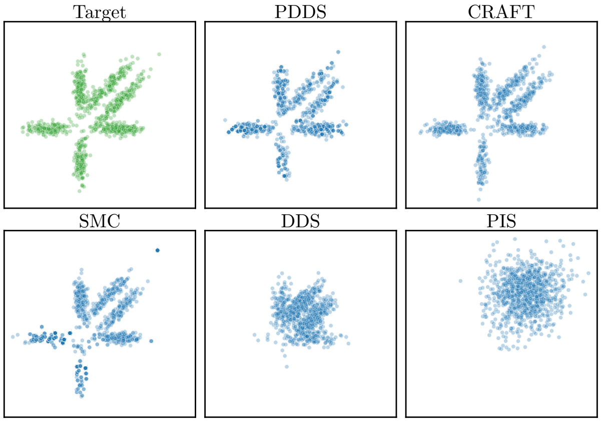

Figure 4: Samples from each method on the Gaussian Mixture task with 4 steps.

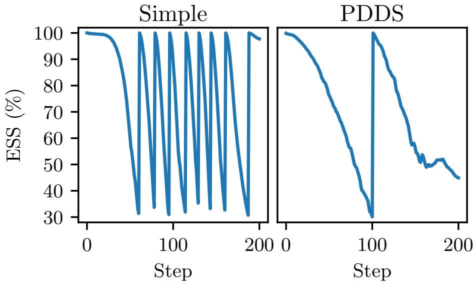

Figure 5: ESS curves on the Sonar task with 200 steps. Left: PDDS with approximation Equation (8), right: PDDS with learnt potential approximation.

duction in number of resampling steps. In the top pane of Figure 7, we see that the simple potential approximation (8) provides very poor normalizing constant estimates. Iterations of Algorithm 2 considerably improve performance.

In the second pane of Figure 7, we show the evolution of $\log \hat{\mathcal{Z}}_0^N$ during training for each of the diffusion-based approaches. While PDDS initially falls below PIS and DDS due to the simple initial approximation, we exceed each of these methods after around 7×10^7 density evaluations.

7. Discussion

This paper contributes to the growing literature on the use of denoising diffusion ideas for Monte Carlo sampling (Berner et al., 2022; McDonald & Barron, 2022; Vargas et al., 2023; Huang et al., 2024; Richter et al., 2024). It proposes an original iterative SMC algorithm which provides an unbiased estimate of the normalizing constant for any finite number of particles by leveraging an original score matching technique. This algorithm also provides asymptotically consistent estimates of the normalizing constant and of expectations with respect to the target. One limitation of PDDS is that it practically relies on $g_0(x)$ being a well-behaved potential function. While our approach using a variational approximation to guide a reparameterization of the target has been effective in our examples, more sophisticated techniques might have to be implemented (Hoffman et al., 2019).

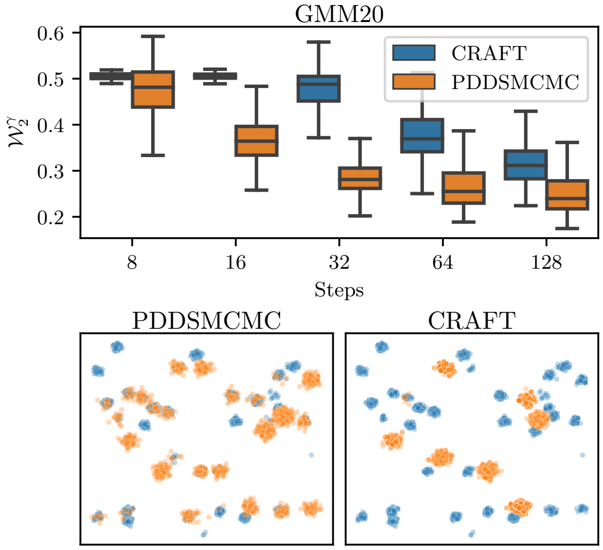

Figure 6: Top: $\mathcal{W}_2^\gamma$ distance between samples from CRAFT and PDDS-MCMC and from GMM20. Variation displayed over training and sampling seeds (200 total). Bottom: samples (first two dimensions) from PDDS-MCMC and CRAFT (orange) using 32 steps versus GMM20 target (blue).

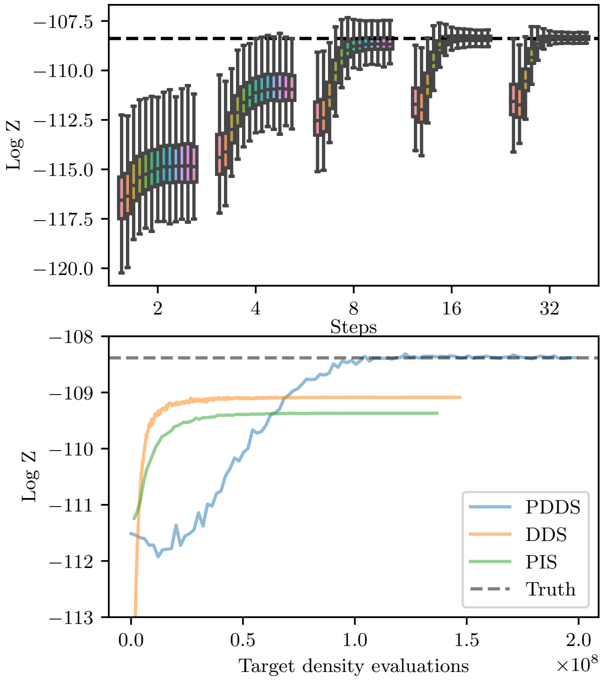

Figure 7: Top: $\log \hat{\mathcal{Z}}_0^N$ every 2 iterations of Algorithm 2 on the Sonar task. Left-most bar of each group shows PDDS with the simple approximation. Bottom: $\log \hat{\mathcal{Z}}_0^N$ during training, one realization on the Sonar task with 16 steps.

Impact Statement

Sampling is a ubiquitous problem. Therefore, PDDS can be applied to a wide range of applications. While we have obtained limit theorems for this scheme, it is important to exercise caution if the output were to be used for decision making as we practically only use a finite number of particles.

Acknowledgements

We would like to thank Pierre Del Moral, Christian A. Naesseth, Sam Power, Saifuddin Syed, Brian Trippe, Francisco Vargas, and Luhuan Wu for helpful discussions. AP is funded by the EPSRC CDT in Modern Statistics and Statistical Machine Learning through grant EP/S023151/1. HD is funded by a postdoctoral position through the CoSInES grant EP/R034710/1. AD acknowledges support from EPSRC grants EP/R034710/1 and EP/R018561/1.

References

Anastasiou, A., Barp, A., Briol, F.-X., Ebner, B., Gaunt, R. E., Ghaderinezhad, F., Gorham, J., Gretton, A., Ley, C., Liu, Q., Mackey, L., Oates, C. J., Reinert, G., and Swan, Y. Stein's method meets computational statistics: a review of some recent developments. *Statistical Science*, 38(1):120–139, 2023.

Arbel, M., Matthews, A., and Doucet, A. Annealed flow transport Monte Carlo. In *International Conference on Machine Learning*, 2021.

Assaraf, R. and Caffarel, M. Zero-variance principle for Monte Carlo algorithms. *Physical Review Letters*, 83: 4682–4685, 1999.

Berner, J., Richter, L., and Ullrich, K. An optimal control perspective on diffusion-based generative modeling. In *NeurIPS Workshop on Score-Based Methods*, 2022.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. JAX: composable transformations of Python+NumPy programs, 2018.

Cardoso, G., Idrissi, Y. J. E., Corff, S. L., and Moulines, E. Monte Carlo guided diffusion for Bayesian linear inverse problems. In *International Conference on Learning Representations*, 2024.

Cattiaux, P., Conforti, G., Gentil, I., and Léonard, C. Time reversal of diffusion processes under a finite entropy condition. *Annales de l'Institut Henri Poincaré (B) Probabilites et statistiques*, 59(4):1844–1881, 2023.

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

Chopin, N. and Papaspiliopoulos, O. *An Introduction to Sequential Monte Carlo*. Springer Ser. Stat. Springer, 2020.

Chopin, N., Singh, S. S., Soto, T., and Vihola, M. On resampling schemes for particle filters with weakly informative observations. *The Annals of Statistics*, 50(6):3197–3222, 2022.

Chung, H., Kim, J., Mccann, M. T., Klasky, M. L., and Ye, J. C. Diffusion posterior sampling for general noisy inverse problems. In *International Conference on Learning Representations*, 2023.

Corso, G., Xu, Y., De Bortoli, V., Barzilay, R., and Jaakkola, T. Particle guidance: non-iid diverse sampling with diffusion models. In *International Conference on Learning Representations*, 2023.

Dai, C., Heng, J., Jacob, P. E., and Whiteley, N. An invitation to sequential Monte Carlo samplers. *Journal of the American Statistical Association*, 117(539):1587–1600, 2022.

De Bortoli, V., Thornton, J., Heng, J., and Doucet, A. Diffusion Schrödinger bridge with applications to score-based generative modeling. In *Advances in Neural Information Processing Systems*, 2021.

Del Moral, P. *Feynman-Kac Formulae: Genealogical and Interacting Particle Approximations*. Springer, 2004.

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

Del Moral, P., Doucet, A., and Jasra, A. On adaptive resampling strategies for sequential Monte Carlo methods. *Bernoulli*, 18(1):252–278, 2012.

Douc, R. and Cappé, O. Comparison of resampling schemes for particle filtering. In *Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis*, pp. 64–69. IEEE, 2005.

Doucet, A., De Freitas, N., and Gordon, N. J. *Sequential Monte Carlo Methods in Practice*. Information Science and Statistics. New York, NY: Springer, New York, 2001.

Gerber, M., Chopin, N., and Whiteley, N. Negative association, ordering and convergence of resampling methods. *The Annals of Statistics*, 47(4):2236–2260, 2019.

Geyer, C. Markov chain Monte Carlo maximum likelihood. In *Computing science and statistics: Proceedings of 23rd Symposium on the Interface Interface Foundation, Fairfax Station, 1991*, pp. 156–163, 1991.

Guarniero, P., Johansen, A. M., and Lee, A. The iterated auxiliary particle filter. *Journal of the American Statistical Association*, 112(520):1636–1647, 2017.

Haussmann, U. G. and Pardoux, E. Time reversal of diffusions. *The Annals of Probability*, 14(3):1188–1205, 1986.

Heng, J., Bishop, A. N., Deligiannidis, G., and Doucet, A. Controlled sequential Monte Carlo. *The Annals of Statistics*, 48(5):2904–2929, 2020.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.

Hoffman, M., Sountsov, P., Dillon, J. V., Langmore, I., Tran, D., and Vasudevan, S. NeuTra-lizing bad geometry in Hamiltonian Monte Carlo using neural transport. *arXiv preprint arXiv:1903.03704*, 2019.

Huang, X., Dong, H., Hao, Y., Ma, Y., and Zhang, T. Reverse diffusion Monte Carlo. In *International Conference on Learning Representations*, 2024.

Hyvärinen, A. Estimation of non-normalized statistical models by score matching. *The Journal of Machine Learning Research*, 6:695–709, 2005.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Kloeden, P. E. and Platen, E. *Numerical Solution of Stochastic Differential Equations*, volume 23 of *Appl. Math. (N. Y.)*. Berlin: Springer-Verlag, 1992.

Lai, C.-H., Takida, Y., Murata, N., Uesaka, T., Mitsufuji, Y., and Ermon, S. Regularizing score-based models with score Fokker-Planck equations. In *NeurIPS Workshop on Score-Based Methods*, 2022.

Lawson, D., Raventós, A., and Linderman, S. SIXO: Smoothing inference with twisted objectives. *Advances in Neural Information Processing Systems*, 2022.

Liptser, R. S. and Shiryayev, A. N. *Statistics of Random Processes. I. General theory. Translated by A. B. Aries*, volume 5 of *Appl. Math. (N. Y.)*. Springer, New York, 1977.

Máté, B. and Fleuret, F. Learning deformation trajectories of Boltzmann densities. *Transactions on Machine Learning Research*, 2023.

Matthews, A. G. D. G., Arbel, M., Rezende, D. J., and Doucet, A. Continual repeated annealed flow transport Monte Carlo. In *International Conference on Machine Learning*, 2022.

McDonald, C. J. and Barron, A. R. Proposal of a score based approach to sampling using Monte Carlo estimation of score and oracle access to target density. In *NeurIPS Workshop on Score-Based Methods*, 2022.

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

Mira, A., Solgi, R., and Imparato, D. Zero variance Markov chain Monte Carlo for Bayesian estimators. *Statistics and Computing*, 23(5):653–662, 2013.

Møller, J., Syversveen, A. R., and Waagepetersen, R. P. Log Gaussian Cox processes. *Scandinavian Journal of Statistics*, 25(3):451–482, 1998.

Neal, R. M. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139, 2001.

Neal, R. M. Slice sampling. *The Annals of Statistics*, 31: 705–767, 06 2003.

Nichol, A. Q. and Dhariwal, P. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, 2021.

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

Salimans, T. and Ho, J. Should EBMs model the energy or the score? In *Energy Based Models Workshop-ICLR 2021*, 2021.

Song, J., Vahdat, A., Mardani, M., and Kautz, J. Pseudoinverse-guided diffusion models for inverse problems. In *International Conference on Learning Representations*, 2023.

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

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

Sountsov, P., Radul, A., and contributors. Inference gym, 2020. URL https://pypi.org/project/inference_gym.

Syed, S., Bouchard-Côté, A., Deligiannidis, G., and Doucet, A. Non-reversible parallel tempering: A scalable highly parallel MCMC scheme. *Journal of the Royal Statistical Society Series B*, 84(2):321–350, 2022.

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

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

Vincent, P. A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7):1661–1674, 2011.

Webber, R. J. Unifying sequential Monte Carlo with resampling matrices. *arXiv preprint arXiv:1903.12583*, 2019.

Woodard, D. B., Schmidler, S. C., and Huber, M. Sufficient conditions for torpid mixing of parallel and simulated tempering. *Electronic Journal of Probability*, 14:780–804, 2009.

Wu, L., Trippe, B. L., Naesseth, C. A., Blei, D., and Cunningham, J. P. Practical and asymptotically exact conditional sampling in diffusion models. In *Advances in Neural Information Processing Systems*, 2023.

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

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

Appendix

The Appendix is organized as follows. In Appendix A we detail the PDDS algorithm when adaptive resampling is used. In Appendix B we reformulate the results of Webber (2019) in terms of chi-squared divergences. In Appendix C, we expose our proofs. Finally, in Appendix D we present details and additional work relating to our experiments.

A. Particle Denoising Diffusion Sampler with Adaptive Resampling

Algorithm 3 Particle Denoising Diffusion Sampler with Adaptive Resampling

```
Input: Schedule $(\beta_t)_{t\in[0,T]}$ as in (2); Approximations $(\hat{g}_k)_{k=0}^K$ s.t. $\hat{g}_0 = g_0, \hat{g}_K = 1$; Number of particles $N$; ESS
    resampling threshold $\alpha$
  Sample $X_K^i \overset{\text{iid}}{\sim} \mathcal{N}(0, \mathrm{I})$ for $i \in [N]$
  Set $\hat{\mathcal{Z}}_K \leftarrow 1$ and $\omega_K^i \leftarrow 1/N$ for $i \in [N]$
  for $k = K-1, \ldots, 0$ do
    Move. Sample $\tilde{X}_k^i \sim \hat{\pi}(\cdot|X_{k+1}^i)$ for $i \in [N]$ (see (16))
    Weight. $\omega_k^i \leftarrow \omega_k^i \, \omega_k(\tilde{X}_k^i, X_{k+1}^i)$ for $i \in [N]$ (see (17))
    Set $\hat{\mathcal{Z}}_k \leftarrow \hat{\mathcal{Z}}_{k+1} \times \frac{1}{N} \sum_{i\in[N]} \omega_k^i$
    Normalize $\omega_k^i \leftarrow \omega_k^i / \sum_{j\in[N]} \omega_k^j$
    Resample and MCMC.
    if $(\sum_{i\in[N]} (\omega_k^i)^2)^{-1} < \alpha N$ then
      $X_k^{1:N} \leftarrow \mathrm{resample}(\tilde{X}_k^{1:N}, \omega_k^{1:N})$ (see Section 3.3)
      (Optional). Sample $X_k^i \leftarrow \mathfrak{M}_k(X_k^i, \cdot)$ for $i \in [N]$ using a $\hat{\pi}_k$-invariant MCMC kernel.
      Reset $\omega_k^{1:N} \leftarrow 1/N$
    else
      $X_k^{1:N} \leftarrow \tilde{X}_k^{1:N}$
    end if
  end for
Output: Particle estimates $\hat{\pi}^N = \frac{1}{N} \sum_{i\in[N]} \delta_{X_0^i}$ of $\pi$ and $\hat{\mathcal{Z}}_0^N$ of $\mathcal{Z}$
```

B. Asymptotic Error Formulae for SMC

The results of Webber (2019) play a fundamental role in our work. Compared to traditional SMC literature, they put more stress on the normalizing constant estimate, have results for different resampling schemes, and do not require weight boundedness.

In this section, we present these results using the language of chi-squared divergence. This gives alternative expressions which are easier to manipulate in our context.

B.1. Chi-squared divergence

For two probability distributions p and q such that $q \ll p$, the chi-squared divergence of q with respect to p is defined as

$$\chi^2(q||p) = \int_{\mathcal{X}} p(\mathrm{d}x) \left[\left(\frac{\mathrm{d}q}{\mathrm{d}p} \right)^2 (x) - 1 \right].$$

We state without proof two simple properties of this divergence.

Lemma B.1. *Let g be a nonnegative function from $\mathcal{X}$ to $\mathbb{R}$ such that $p(g) := \mathbb{E}_p[g(X)] < \infty$. Define the probability distribution π by $\pi(\mathrm{d}x) \propto p(x)g(x)$. Then*

$$\mathrm{Var}_p(g) = (p(g))^2 \, \chi^2(\pi||p).$$

Lemma B.2. *The chi-squared divergence between two probability distributions* p *and* q *on* $\mathcal{X} \times \mathcal{Y}$ *satisfies the decomposition*

$$\chi^2(q(\mathrm{d}x, \mathrm{d}y)||p(\mathrm{d}x, \mathrm{d}y)) = \chi^2(q(\mathrm{d}x)||p(\mathrm{d}x)) + \int p(\mathrm{d}x) \left(\frac{\mathrm{d}q}{\mathrm{d}q}\right)^2 (x)\chi^2(q(\mathrm{d}y|x)||p(\mathrm{d}y|x)).$$

B.2. Generic Feynman-Kac formula and the associated SMC algorithm

Let

$$\mathbb{M}(\mathrm{d}x_{0:T}) = \mathbb{M}(\mathrm{d}x_0)\mathbb{M}(\mathrm{d}x_1|x_0)\dots\mathbb{M}(\mathrm{d}x_T|x_{T-1})$$

be a Markov measure defined on $\mathcal{X}_0 \times \ldots \times \mathcal{X}_T$. Let $G_0(x_0)$, $G_1(x_0, x_1)$, …, $G_T(x_{T-1}, x_T)$ be strictly positive functions. For any $t < T$, assume that there exists $Z_t > 0$ such that

$$\mathbb{Q}_t(\mathrm{d}x_{0:T}) = \frac{1}{Z_t}\mathbb{M}_0(\mathrm{d}x_0)G_0(x_0)\mathbb{M}_1(\mathrm{d}x_1|x_0)G_1(x_0, x_1)\dots\mathbb{M}_t(\mathrm{d}x_t|x_{t-1})G_t(x_t)\mathbb{M}(\mathrm{d}x_{t+1:T}|x_t)$$

is a probability measure. Then $\mathbb{Q}_t(\mathrm{d}x_{0:t})$ is called the Feynman-Kac model associated with the Markov measure $\mathbb{M}(\mathrm{d}x_{0:t})$ and the weight functions $(G_s)_{s\leq t}$. Given a number of particles N, Algorithm 4 approximates $\mathbb{Q}_T(\mathrm{d}x_T)$ and the normalizing constant Z_T. It is called the SMC algorithm associated to the Feynman-Kac model $\mathbb{Q}_T$.

Algorithm 4 Generic SMC algorithm

```
Input: Markov kernels $\mathbb{M}(\mathrm{d}x_t|x_{t-1})$; Functions $G_t(x_{t-1}, x_t)$; Number of particles $N$
  Sample $X_0^{1:N} \overset{\text{iid}}{\sim} \mathbb{M}_0(\mathrm{d}x_0)$
  Set $\omega_0^n = G_0(X_0^n)$ and $Z_0^N = \frac{1}{N}\sum \omega_0^n$
  Normalize $\omega_0^n \leftarrow \omega_0^n / \sum_m \omega_0^m$
  for $t = 1, \dots, T$ do
    Resample particles $\tilde{X}_{t-1}^{1:N}$ among particles $X_{t-1}^{1:N}$ with weights $\omega_{t-1}^{1:N}$
    Sample $X_t^n \sim \mathbb{M}_t(\mathrm{d}x_t|\tilde{X}_{t-1}^n)$
    Set $\omega_t^n = G_t(\tilde{X}_{t-1}^n, X_t^n)$ and $Z_t^n = Z_{t-1}^n \frac{1}{N}\sum \omega_t^n$
    Normalize $\omega_t^n \leftarrow \omega_t^n / \sum \omega_t^m$
  end for
Output: Empirical measure $\sum \omega_T^n \delta_{X_T^n}$ approximating $\mathbb{Q}_T(\mathrm{d}x_T)$ and estimate $Z_T^N$ approximating $Z_T$
```

B.3. Sorted stratified resampling schemes

The resampling step of Algorithm 4 can be done in a number of different ways. It is well known that multinomial resampling should be avoided. Practitioners often rely on alternative schemes, in particular systematic resampling. However, the theoretical properties of these schemes are less well-studied.

To investigate theoretically the asymptotic error of PDDS when the discretization step tends to zero, we consider the stratified resampling scheme where particles are sorted by a certain coordinate $\theta : \mathbb{R}^d \to \mathbb{R}$, see Algorithm 5.

Algorithm 5 Generic sorted stratified resampling

```
Input: Particles $X^{1:N}$ in $\mathbb{R}^d$ with weights $W^{1:N}$; Sorting function $\theta : \mathbb{R}^d \to \mathbb{R}$
  Sort the particles $X^{1:N}$ such that $\theta(X^{s_1}) \leq \ldots \leq \theta(X^{s_N})$ for a permutation $s_{1:N}$ of $\{1, 2, \ldots, N\}$
  for $n = 1, \dots, N$ do
    Simulate $U^n \sim \text{Uniform}[0, 1]$
    Find the index $k_n$ such that $\sum_{i=1}^{k_n-1} W^{s_i} \leq \frac{n-1+U^n}{N} < \sum_{i=1}^{k_n} W^{s_i}$
    Set $\tilde{X}_n \leftarrow X^{s_{k_n}}$
  end for
Output: Resampled particles $\tilde{X}^{1:N}$
```

Depending on the chosen coordinate θ, sorted stratified resampling can significantly reduce the asymptotic error of the particle filter. Precise formulations is given in Webber (2019, Theorem 3.2). In a nutshell, the variance of Z_T^N/Z_T comes from two sources: the mismatch between the PF proposal $\mathbb{M}_t(\mathrm{d}x_t|x_{t-1})$ and the target law $\mathbb{Q}_T(\mathrm{d}x_t|x_{t-1})$; and the error caused by the resampling step. The first source is common to all resampling methods. The magnitude of the second source for multinomial resampling is $\sum_{t=1}^T \mathrm{Var}_{\mathbb{Q}_{t-1}}\left(\bar{h}_{t-1}(X_{t-1})\right)$ where

$$\bar{h}_{t-1}(x_{t-1}) = \frac{\mathbb{Q}_T(\mathrm{d}x_{t-1})}{\mathbb{Q}_{t-1}(\mathrm{d}x_{t-1})} = \frac{\int \prod_{s=t}^T \mathbb{M}_s(\mathrm{d}x_s|x_{s-1}) G_s(x_{s-1},x_s)\mathrm{d}x_{t:T}}{\int \mathbb{Q}_{t-1}(\mathrm{d}x'_{t-1}) \prod_{s=t}^T \mathbb{M}_s(\mathrm{d}x_s|x_{s-1}) G_s(x_{s-1},x_s)\mathrm{d}x'_{t-1}\mathrm{d}x_{t:T}}.$$

Write the resampling error for multinomial resampling as

$$\sum_{t=1}^T \mathrm{Var}_{\mathbb{Q}_{t-1}}\left(\bar{h}_{t-1}(X_{t-1})\right) = \sum_{t=1}^T \mathbb{E}_{\mathbb{Q}_{t-1}}\left[\mathrm{Var}_{\mathbb{Q}_{t-1}}\left(\bar{h}_{t-1}(X_{t-1})|\theta(X_{t-1})\right)\right] + \sum_{t=1}^T \mathrm{Var}_{\mathbb{Q}_{t-1}}\left(\mathbb{E}_{\mathbb{Q}_{t-1}}\left[\bar{h}_{t-1}(X_{t-1})|\theta(X_{t-1})\right]\right).$$

Then the error for stratified resampling when particles are sorted by $\theta : \mathbb{R}^d \to \mathbb{R}$ contains only the first term and not the second one. Thus ideally we would like to choose $\theta = \bar{h}_{t-1}$ so that

$$\mathrm{Var}\left(\bar{h}_{t-1}(X_{t-1})|\theta(X_{t-1})\right) = 0. \tag{21}$$

While the ideal function $\bar{h}_{t-1}$ is intractable, there are more generic choices of θ which guarantee (21). If $\theta : \mathbb{R}^d \to \mathbb{R}$ is an injective map then (21) automatically holds. Such maps usually arise as pseudo-inverses of space-filling curves.

Following Gerber et al. (2019), we take θ to be the pseudo-inverse of the Hilbert curve, of which the existence is given by their Proposition 2. More precisely, that proposition gives an injective map from $[0,1]^d$ to $[0,1]$; so to get an injection from $\mathbb{R}^d$ to $[0,1]$ one might first apply any injection from $\mathbb{R}^d$ to $[0,1]^d$. In practice, numerical implementations are available to sort particles with the Hilbert curve, for instance the `hilbert` module of the Python package `particles` .

We point out that we consider this particular resampling strategy mainly for theoretical convenience. In our experiments, adaptive systematic resampling works well, but as we mentioned earlier, very little is known about their theoretical properties.

B.4. Asymptotic error

We recall the following definition from Webber (2019).

Definition B.3. The notation $|Y_n - c| \lesssim U_n$ means that there exists a sequence of sets (B_n) such that $\mathbb{P}(B_n) \to 1$ and

$$\limsup_{n\to\infty} \mathbb{E}\left[\mathbb{1}_{B_n}\left|\frac{Y_n - c}{U_n}\right|^2\right] \leq 1.$$

We are now ready to restate parts of Theorem 3.2 and Example 3.4 of Webber (2019) in terms of chi-squared divergences.

Theorem B.4. *Given the Feynman-Kac model defined in Section B.2, assume that $\chi^2(\mathbb{Q}_T(\mathrm{d}x_0)||\mathbb{M}_0(\mathrm{d}x_0)) < \infty$ and $\chi^2(\mathbb{Q}_T(\mathrm{d}x_t)||\mathbb{Q}_t(\mathrm{d}x_t)) < \infty$, $\forall t$. Then $\sqrt{N}(Z_T^N/Z_T - 1)$ is asymptotically normal with variance σ^2_{mult} if multinomial resampling is used; $\left|\sqrt{N}(Z_T^N/Z_T - 1)\right|^2 \lesssim \sigma^2_{\mathrm{sort}}$ if sorted resampling (Gerber et al., 2019) is used; with*

$$\begin{aligned}\sigma^2_{\mathrm{mult}} &= \chi^2(\mathbb{Q}_T(\mathrm{d}x_0)||\mathbb{M}_0(\mathrm{d}x_0)) + \sum_{t=1}^T \chi^2(\mathbb{Q}_T(\mathrm{d}x_{t-1},\mathrm{d}x_t)||\mathbb{Q}_{t-1}(\mathrm{d}x_{t-1})M_t(x_{t-1},\mathrm{d}x_t)) \\ &= \underbrace{\chi^2(\mathbb{Q}_T(\mathrm{d}x_0)||\mathbb{M}_0(\mathrm{d}x_0)) + \sum_{t=1}^T \int \mathbb{Q}_{t-1}(\mathrm{d}x_{t-1})\left(\frac{\mathbb{Q}_T(\mathrm{d}x_{t-1})}{\mathbb{Q}_{t-1}(\mathrm{d}x_{t-1})}\right)^2 \chi^2(\mathbb{Q}_T(\mathrm{d}x_t|x_{t-1})||\mathbb{M}_t(\mathrm{d}x_t|x_{t-1})) +}_{\sigma^2_{\mathrm{sort}}} \\ &\quad + \sum_{t=1}^T \chi^2(\mathbb{Q}_T(\mathrm{d}x_{t-1})||\mathbb{Q}_{t-1}(\mathrm{d}x_{t-1}))\end{aligned}$$

where we have decomposed σ^2_{mult} using the chain rule (Lemma B.2).

Proof. The original formulation of Theorem 3.2 (Webber, 2019) is written in terms of the following quantities

$$\tilde{G}_t := \mathbb{E}_{\mathbb{M}}\left[\prod_{s=0}^{t-1} G_s\right] G_t / \mathbb{E}\left[\prod_{s=0}^{t} G_s\right],$$

$$h_t(x_t) := \mathbb{E}\left[\prod_{s=t+1}^{T} G_s \middle| X_t = x_t\right].$$

To translate these notations into our case, note that $\tilde{G}_t = \mathbb{Q}_t(\mathrm{d}x_{0:T})/\mathbb{Q}_{t-1}(\mathrm{d}x_{0:T})$ and thus

$$\min_{c\in\mathbb{R}} \mathbb{E}_{\mathbb{M}}\left[\prod_{s=0}^{t} \tilde{G}_s |h_t - c|^2\right] = \min_{c\in\mathbb{R}} \mathbb{E}_{\mathbb{Q}_t}\left[|h_t - c|^2\right] = \mathrm{Var}_{\mathbb{Q}_t}(h_t) = [\mathbb{Q}_t(h_t)]^2\, \chi^2(\mathbb{Q}_T(\mathrm{d}x_t)||\mathbb{Q}_t(\mathrm{d}x_t))$$

using Lemma B.1. Moreover,

$$\mathrm{Var}_{\mathbb{M}}(G_{t+1}(X_t, X_{t+1})h_{t+1}(X_{t+1})|X_t) = h_t^2(X_t)\chi^2(\mathbb{Q}_T(\mathrm{d}x_{t+1}|x_t)||\mathbb{M}_T(\mathrm{d}x_{t+1}|x_t))$$

and thus, using $h_t(x_t) = \frac{\mathbb{Q}_T(\mathrm{d}x_t)}{\mathbb{Q}_t(\mathrm{d}x_t)}\mathbb{Q}_t(h_t)$ we get

$$\mathbb{E}\left[\prod_{s=0}^{t} G_s \,\mathrm{Var}(G_{t+1}h_{t+1}|X_t)\right] = Z_t\mathbb{Q}_t(h_t)^2\mathbb{E}_{\mathbb{Q}_t}\left[\left\{\frac{\mathbb{Q}_T(\mathrm{d}x_t)}{\mathbb{Q}_t(\mathrm{d}x_t)}(X_t)\right\}^2 \chi^2(\mathbb{Q}_T(\mathrm{d}x_{t+1}|X_t)||\mathbb{M}_T(\mathrm{d}x_{t+1}|X_t))\right].$$

The identity $Z_t\mathbb{Q}_t(h_t) = Z_T$ helps conclude the proof. □

C. Proofs

C.1. Proof of Proposition 2.1

Proof. Write

$$\begin{aligned}\pi_t(x_t) &= \int \pi_0(x_0)p(x_t|x_0)\mathrm{d}x_0 = \frac{1}{\mathcal{Z}}\int g_0(x_0)p_0(x_0)p(x_t|x_0)\mathrm{d}x_0 \\ &= \frac{1}{\mathcal{Z}}\int g_0(x_0)p_0(x_t)p(x_0|x_t)\mathrm{d}x_0 \\ &= \frac{1}{\mathcal{Z}}p_0(x_t)g_t(x_t).\end{aligned}\tag{22}$$

The proof is concluded by taking the log gradient of the obtained identity with respect to x_t. □

C.2. Proof of Lemma 2.2

We first state the following elementary lemma on the solution of a linear SDE (Kloeden & Platen, 1992, Chapter 4.2).

Lemma C.1. *Let $a : [0, T] \to \mathbb{R}$ and $c : [0, T] \to \mathbb{R}$ be two continuous functions. Put $\Phi_t = \exp\int_0^t a_s\mathrm{d}s$, $\forall t \in [0, T]$. Then the solution to*

$$\mathrm{d}Z_t = (a_tZ_t + c_t)\mathrm{d}t + \sqrt{2}\mathrm{d}W_t \tag{23}$$

is

$$Z_t = \Phi_t\left(Z_0 + \int_0^t \Phi_s^{-1}c_s\mathrm{d}s + \int_0^t \Phi_s^{-1}\sqrt{2}\mathrm{d}W_s\right). \tag{24}$$

Using Itô isometry, we get the following straightforward corollary.

Corollary C.2. *Under the setting of Lemma C.1, if $Z_0 \sim \mathcal{N}(0, 1)$, then*

$$\mathbb{E}[Z_t] = \Phi_t\int_0^t \Phi_s^{-1}c_s\mathrm{d}s, \quad \mathrm{Var}(Z_t) = \Phi_t^2 + 2\Phi_t^2\int_0^t \Phi_s^{-2}\mathrm{d}s. \tag{25}$$

We are now ready to give the proof of Proposition 2.2.

Proof of Proposition 2.2. Without loss of generality, we can assume that $\beta_t \equiv 1$. Indeed, putting $\hat{\bar{g}}_t(x_t) := g_0(e^{-t}x_t)$ and defining $\bar{Z}_t^{(T)}$ by

$$\mathrm{d}\bar{Z}_t^{(T)} = \left[-\bar{Z}_t^{(T)} + 2\nabla \log \hat{\bar{g}}_{T-t}(\bar{Z}_t^{(T)})\right] \mathrm{d}t + \sqrt{2}\mathrm{d}\tilde{B}_t, \quad \bar{Z}_0^{(T)} \sim \mathcal{N}(0, 1), \tag{26}$$

we see that $Z_{0:t}^{(T)}$ has the same law as $\bar{Z}^{\int_0^T \beta_s \mathrm{d}s}_{\int_{T-t}^T \beta_s \mathrm{d}s}$.

We only consider the case $\sigma \neq 1$ here. The case $\sigma = 1$ can be treated using similar but simpler calculations; it can also be recovered by letting $\sigma \to 1$ in the expressions below.

Since $\nabla \log g_0(x_0) = \nabla \log \pi(x_0) - \nabla \log p_0(x_0) = -(x - \mu)/\sigma^2 + x$, Equation (9) has the form (23) with

$$a_t = -\left[1 + 2e^{2(t-T)}(1/\sigma^2 - 1)\right], \quad c_t = 2e^{t-T}\mu/\sigma^2. \tag{27}$$

Tedious but standard calculations yield

$$\Phi_t = \exp\big\{-t - e^{-2T}(1/\sigma^2 - 1)(e^{2t} - 1)\big\} \tag{28}$$

and the integral of t-dependent terms in $\int_0^T \Phi_t^{-1} c_t \mathrm{d}t$ is

$$\int_0^T \exp\Big\{2t + (1/\sigma^2 - 1)e^{2(t-T)}\Big\}\mathrm{d}t = \frac{1}{2e^{-2T}(1/\sigma^2 - 1)}\left[\exp\big\{1/\sigma^2 - 1\big\} - \exp\big\{e^{-2T}(1/\sigma^2 - 1)\big\}\right].$$

Using Corollary C.2, we have

$$\mathbb{E}[Z_T^{(T)}] = \frac{\mu}{1 - \sigma^2}\left[1 - \exp\big\{(e^{-2T} - 1)(1/\sigma^2 - 1)\big\}\right] \tag{29}$$

and $\mathrm{Var}(Z_T^{(T)}) = v_{T,1} + v_{T,2}$, where

$$v_{T,1} = \frac{1}{2(1/\sigma^2 - 1)}\left[1 - \exp\big\{-2(1/\sigma^2 - 1)(1 - e^{-2T})\big\}\right], \tag{30}$$

$$v_{T,2} = \exp\big\{-2T - 2(1/\sigma^2 - 1)(1 - e^{-2T})\big\}. \tag{31}$$

The lemma is proved. □

C.3. Proof of Propositions 3.1 and 3.2

The propositions follow from a direct application of Theorem B.4 to the Feynman-Kac model

$$\mathbb{Q}_K(y_{0:K}) = \frac{1}{\mathcal{Z}}\mathbb{M}_K(y_{0:K})G_0(y_0)\prod_{k=1}^{K} G_k(y_{k-1}, y_k)$$

where we have the correspondence $y_k = x_{K-k}$ and

$$\begin{aligned} \mathbb{M}_K(y_{0:K}) &= \mathcal{N}(x_K|0, \mathrm{Id})\hat{\pi}(x_{K-1}|x_K)\ldots\hat{\pi}(x_0|x_1) \\ G_0(y_0) &= \hat{g}_K(x_K) \\ G_k(y_{k-1}, y_k) &= \omega_{K-k}(x_{K-k}, x_{K-k+1}). \end{aligned}$$

C.4. Proof of Proposition 3.3

We start by providing some intuition for the result. Repeat that X_k is a shorthand for $X_{k\delta}$ where δ is the discretization step size. This convention only applies for X_k.

When K is big (δ is small), we have, as established in (14),

$$\pi(x_k|x_{k+1}) \approx \mathcal{N}(x_k; \sqrt{1-\alpha_{k+1}}x_{k+1} + \alpha_{k+1}\nabla \log g_{k+1}(x_{k+1}), \alpha_{k+1}I),$$
$$\hat{\pi}(x_k|x_{k+1}) = \mathcal{N}(x_k; \sqrt{1-\alpha_{k+1}}x_{k+1} + \alpha_{k+1}\nabla \log \hat{g}_{k+1}(x_{k+1}), \alpha_{k+1}I).$$

Using the analytic formula for the chi-squared divergence between two Gaussians we get

$$\chi^2(\pi(x_k|x_{k+1})||\hat{\pi}(x_k|x_{k+1})) \approx e^{\alpha_{k+1}\|\nabla \log g_{k+1} - \nabla \log \hat{g}_{k+1}\|^2(x_{k+1})} - 1 \approx 2\delta \|\nabla \log g_{k+1} - \nabla \log \hat{g}_{k+1}\|^2(x_{k+1}).$$

Thus

$$\begin{aligned}
\zeta_K^2 &= \chi^2(\pi_K||\mathcal{N}(0,\mathrm{Id})) + \sum_k \int \frac{\pi_{k+1}(x_{k+1})^2}{\hat{\pi}_{k+1}(x_{k+1})} \chi^2(\pi(x_k|x_{k+1})||\hat{\pi}(x_k|x_{k+1}))\mathrm{d}x_{k+1} \\
&\approx \chi^2(\pi_K||\mathcal{N}(0,\mathrm{Id})) + \sum_k \int \frac{\pi_{k+1}(x_{k+1})^2}{\hat{\pi}_{k+1}(x_{k+1})} 2\delta \|\nabla \log g_{k+1} - \nabla \log \hat{g}_{k+1}\|^2(x_{k+1})\mathrm{d}x_{k+1} \\
&\approx \chi^2(\pi_T||\mathcal{N}(0,\mathrm{Id})) + 2\int_0^T \int_{\mathcal{X}} \frac{\pi_t(x)^2}{\hat{\pi}_t(x)} \|\nabla \log g_t(x) - \nabla \log \hat{g}_t(x)\|^2 \mathrm{d}x\mathrm{d}t.
\end{aligned}$$

C.4.1. Regularity conditions for Proposition 3.3

We assume that the sequence of distributions $\pi_t(\cdot)$ satisfy the following properties.

Assumption C.3. There exists $M_1 > 0$ such that $\|\nabla \log \pi_t(x_t)\| \leq M_1(1 + \|x_t\|)$.

Assumption C.4. There exist $M_2 > 0$, $M_3 > 0$, $\alpha_2 \geq 1$, and $\alpha_3 \geq 1$ such that $\left\|\nabla^2 \log \pi_t(x_t)\right\| \leq M_2(1 + \|x_t\|)^{\alpha_2}$ and $\left\|\nabla^3 \log \pi_t(x_t)\right\| \leq M_3(1 + \|x_t\|)^{\alpha_3}$.

Assumption C.5. There exist $\vartheta > 0$ and $M_\infty < \infty$ such that $\int \pi_t(x_t)e^{\vartheta\|x_t\|^2}\mathrm{d}x_t < M_\infty, \forall t$.

These assumptions are satisfied, for example, when the target distribution π_0 is Gaussian.

Remark C.6. Let $\varphi(x_t) = \nabla \log \pi_t(x_t)$. Then the notation $\nabla^3 \log \pi_t(x_t)$ refers to the second order differential $\varphi''(x_t)$ which is a bilinear mapping from $\mathbb{R}^d \times \mathbb{R}^d$ to $\mathbb{R}^d$. For any multilinear operator $H : \mathcal{X}_1 \times \ldots \times \mathcal{X}_n \to \mathcal{Y}$, we define

$$\|H\|_{\mathrm{op}} := \sup_{x_1,\ldots,x_n \neq 0} \frac{\|H(x_1,\ldots,x_n)\|}{\|x_1\|\ldots\|x_n\|}.$$

By writing $\|H\|$ we implicitly refer to $\|H\|_{\mathrm{op}}$. In fact, the space of such operators is of finite dimensions in our cases of interests, hence any two norms are bounded by a constant factor of each other.

The above assumptions only concern the differential of $\nabla \log \pi_t(x_t)$ with respect to x. The following lemma derives a bound with respect to t.

Lemma C.7. *Under the above assumptions, there exist constants $\bar{M}_1$ and $\bar{\alpha}_1$ such that for all t*

$$\left\|\frac{\partial}{\partial t}\nabla \log \pi_t(x_t)\right\| \leq \bar{M}_1(1 + \|x_t\|)^{\bar{\alpha}_1}.$$

Proof. Using the Fokker–Planck equation for the score (Lai et al., 2022), we write

$$\partial_t \log g_t(x_t) = \mathrm{div}_x \nabla \log g_t(x_t) + \nabla \log g_t(x_t) \circ (\nabla \log g_t(x_t) - x_t).$$

For a fixed t, put $\varphi(x_t) = \nabla \log g_t(x_t)$ and $\psi(x_t) = \mathrm{Tr}(\varphi'(x_t)) + \varphi(x_t) \circ (\varphi(x_t) - x_t)$. Then $\partial_t \nabla \log g_t(x_t) = \nabla\psi(x_t)$. Viewing $\psi'(x_t)$, $\varphi'(x_t)$, and $\varphi''(x_t)$ as elements of $\mathcal{L}(\mathbb{R}^d, \mathbb{R})$, $\mathcal{L}(\mathbb{R}^d, \mathbb{R}^d)$, and $\mathcal{L}(\mathbb{R}^d, \mathcal{L}(\mathbb{R}^d, \mathbb{R}^d))$ respectively; where $\mathcal{L}(A, B)$ is the space of linear operators from A to B; we can write

$$\psi'(x_t)h = \mathrm{Tr}(\varphi''(x_t)h) + (\varphi'(x_t)h) \circ (\varphi(x_t) - x_t) + \varphi(x_t) \circ (\varphi'(x_t)h - h), \forall h \in \mathbb{R}^d$$

where $\circ$ stands for the usual scalar product between two vectors in $\mathbb{R}^d$. There is a constant C depending on the dimension such that $\mathrm{Tr}(L) \leq C\|L\|_{\mathrm{op}}$ for endomorphisms L. Thus

$$\begin{aligned}
\|\psi'(x_t)h\|_{\mathrm{op}} &\leq C\|\varphi''(x_t)\|_{\mathrm{op}}\|h\| + \|\varphi'(x_t)\|_{\mathrm{op}}\|\varphi(x_t) - x_t\|\|h\| + \|\varphi(x_t)\|\|\varphi'(x_t) - \mathrm{Id}\|_{\mathrm{op}}\|h\| \\
&\leq \bar{M}_1(1 + \|x_t\|)^{\bar{\alpha}_1}\|h\|
\end{aligned}$$

for some $\bar{M}_1$ and $\bar{\alpha}_1$ by Assumptions C.3 and C.4. This entails $\|\partial_t \nabla \log g_t(x_t)\| = \|\psi'(x_t)\|_{\mathrm{op}} \leq \bar{M}_1(1 + \|x_t\|)^{\bar{\alpha}_1}$. □

C.4.2. Formal proof

To formalize the error of the heuristic approximations presented at the beginning of Section C.4 we need the following technical lemma.

Lemma C.8. *Let x_0 and v be two vectors in $\mathbb{R}^d$ and suppose that X_t^{A} and X_t^{B} are respectively the unique solutions of the SDEs:*

$$\begin{aligned} &(\mathrm{A}): \quad \mathrm{d}X_t^{\mathrm{A}} = (-X_t^{\mathrm{A}} + 2v)\mathrm{d}t + \sqrt{2}\mathrm{d}W_t^{\mathrm{A}}, \quad X_0 = x_0 \\ &(\mathrm{B}): \quad \mathrm{d}X_t^{\mathrm{B}} = (-X_t^{\mathrm{B}} + 2f_t(X_t^{\mathrm{B}}))\mathrm{d}t + \sqrt{2}\mathrm{d}W_t^{\mathrm{B}}, \quad X_0 = x_0 \end{aligned}$$

where there exist strictly positive constants M_1, $\bar{M}_1$, M_2, and M_3; and strictly greater than 1 constants $\bar{\alpha}_1$, α_2, and α_3; such that the function $f_t(x_t)$ satisfies $\|f_t(x_t)\| \leq M_1(1 + \|x_t\|)$, $\|\partial_t f_t(x_t)\| \leq \bar{M}_1(1 + \|x_t\|)^{\bar{\alpha}_1}$, and $\left\|\nabla^i f_t(x_t)\right\| \leq M_{i+1}(1 + \|x_t\|)^{\alpha_{i+1}}$ for $i \in \{1, 2\}$. (The notation ∇ refers implicitly to the gradient with respect to x.) Denote $\mathbb{P}_{\mathrm{A}}(\mathrm{d}x_{[0,T]})$ and $\mathbb{P}_{\mathrm{B}}(\mathrm{d}x_{[0:T]})$ respectively the path measures associated with the solutions of (A) and (B). Then, there exist a parameter $\widetilde{M}$ depending on all the aforementioned constants and a parameter $\widetilde{M}_1$ depending only on M_1 such that for any $t \leq 1/\widetilde{M}_1$, the chi-squared divergence of $\mathbb{P}_{\mathrm{B}}(\mathrm{d}x_{[0,t]})$ with respect to $\mathbb{P}_{\mathrm{A}}(\mathrm{d}x_{[0,t]})$ is finite, and

$$\left|\chi^2(\mathbb{P}_{\mathrm{B}}(\mathrm{d}x_{[0:t]})||\mathbb{P}_{\mathrm{A}}(\mathrm{d}x_{[0:t]})) - 2t\|f_0(x_0) - v\|^2\right| \leq \widetilde{M}t^2 e^{t\widetilde{M}_1(\|x_0\|^2 + \|v\|^2)}\left(1 + \|x_0\| + \|v\|\right)^{4(1+\bar{\alpha}_1+\alpha_2+\alpha_3)}$$

Proof. Put $\Delta_t(x_t) = 2f_t(x_t) - 2v$. By an application of Girsanov's theorem (Liptser & Shiryayev, 1977, Example 3, Section 6.2.3), we have

$$D_t := \frac{\mathrm{d}\mathbb{P}_{\mathrm{B}}}{\mathrm{d}\mathbb{P}_{\mathrm{A}}}(X_{[0,t]}) = \exp\left\{\int_0^t \frac{\langle \Delta_s(X_s), \mathrm{d}W_s^{\mathrm{A}}\rangle}{\sqrt{2}} - \frac{1}{4}\int_0^t \|\Delta_s(X_s)\|^2 \mathrm{d}s\right\}$$

where, for a vector-valued process V_t, the notation $\int_0^t \langle V_s, \mathrm{d}W_s\rangle := \sum_{i=1}^d \int_0^t V_s^i \mathrm{d}W_s^i$. As a preliminary step, we would like to bound $\mathbb{E}_{\mathrm{A}}[D_t^{\alpha}(1 + \|X_t\| + \|v\|)^n]$ for some $\alpha, n \geq 1$. Here, $c(\cdot)$ denotes a constant whose value might change from line to line and depends on the variables inside the round bracket. We also drop the subscript/superscript A from $\mathbb{E}_{\mathrm{A}}$ and W_t^{A} whenever there is no risk of confusion. We have

$$\begin{aligned} &\mathbb{E}[D_t^{\alpha}(1 + \|X_t\| + \|v\|)^n] = \mathbb{E}\left[\exp\left\{\alpha\int_0^t \frac{\langle \Delta_s, \mathrm{d}W_s\rangle}{\sqrt{2}} - \frac{\alpha}{4}\int_0^t \|\Delta_s\|^2 \mathrm{d}s\right\}(1 + \|X_t\| + \|v\|)^n\right] \\ &= \mathbb{E}\left[\exp\left\{\alpha\int_0^t \frac{\langle \Delta_s, \mathrm{d}W_s\rangle}{\sqrt{2}} - \frac{\alpha^2}{2}\int_0^t \|\Delta_s\|^2 \mathrm{d}s\right\}\exp\left\{\left(\frac{\alpha^2}{2} - \frac{\alpha}{4}\right)\int_0^t \|\Delta_s\|^2 \mathrm{d}s\right\}(1 + \|X_t\| + \|v\|)^n\right] \\ &\leq e^{t(c(M_1,\alpha) + c(\alpha)\|v\|^2)}\mathbb{E}\Bigg[\exp\left\{\alpha\int_0^t \frac{\langle \Delta_s, \mathrm{d}W_s\rangle}{\sqrt{2}} - \frac{\alpha^2}{2}\int_0^t \|\Delta_s\|^2 \mathrm{d}s\right\}\exp\left\{c(M_1, \alpha)\int_0^t \|X_s\|^2 \mathrm{d}s\right\}\times \\ &\times (1 + \|X_t\| + \|v\|)^n\Bigg] \text{ using } \|\Delta_s\|^2 \leq c(M_1)(1 + \|x_s\|^2) + 8\|v\|^2 \\ &\leq e^{t(c(M_1,\alpha) + c(\alpha)\|v\|^2)}\mathbb{E}^{1/2}\left[\exp\left\{\sqrt{2}\alpha\int_0^t \langle \Delta_s, \mathrm{d}W_s\rangle - \alpha^2\int_0^t \|\Delta_s\|^2 \mathrm{d}s\right\}\right]\times \\ &\times \mathbb{E}^{1/4}\left[\exp\left\{4c(M_1, \alpha)\int_0^t \|X_s\|^2 \mathrm{d}s\right\}\right]\mathbb{E}^{1/4}\left[(1 + \|X_t\| + \|v\|)^n\right] \end{aligned}$$

using double Cauchy-Schwarz $\mathbb{E}[XYZ] \leq \mathbb{E}^{1/2}[X^2]\mathbb{E}^{1/4}[Y^4]\mathbb{E}^{1/4}[Z^4]$.

In the last line of the above display, the first expectation is equal to 1 by the same Girsanov argument as Liptser & Shiryayev (1977, Example 3, Section 6.2.3). To bound the second expectation, we note that under $\mathbb{P}_{\mathrm{A}}$, we have $X_s \sim \mathcal{N}(\sqrt{1 - \lambda_s}(x_0 - 2v) + 2v, \lambda_s)$. Elementary calculations yield the bound

$$\mathbb{E}[e^{k\|X_s\|^2}] = \left(\frac{1}{\sqrt{1 - 2k\lambda_s}}\right)^d \exp\left\{\frac{k\left\|\sqrt{1 - \lambda_s}(x_0 - 2v) + 2v\right\|^2}{1 - 2k\lambda_s}\right\} \leq (\sqrt{2})^d e^{16k(\|x_0\|^2 + \|v\|^2)}$$

for $0 < k < 1/4$. Write

$$\mathbb{E}\left[\exp\left\{4c(M_1,\alpha)\int_0^t \|X_s\|^2 \mathrm{d}s\right\}\right] \le \frac{1}{t}\int_0^t \mathbb{E}\left[e^{4tc(M_1,\alpha)\|X_s\|^2}\right]\mathrm{d}s \le c(1)e^{64tc(M_1,\alpha)(\|x_0\|^2+\|v\|^2)}$$

if $t \le \frac{1}{16c(M_1,\alpha)}$, using Jensen's inequality and the above bound. The third expectation can be bounded by $c(n)(1+\|x_0\|^{4n}+\|v_0\|^{4n})$.

Putting everything together, we establish that there exist constants $c(n)$ and $c(M_1,\alpha)$ such that

$$\mathbb{E}[D_t^\alpha(1+\|X_t\|+\|v\|)^n] \le c(n)e^{tc(M_1,\alpha)(\|x_0\|^2+\|v\|^2)}(1+\|x_0\|^{4n}+\|v\|^{4n}), \forall t \le \frac{1}{c(M_1,\alpha)}. \tag{32}$$

Now to study $\chi^2(\mathbb{P}_\mathrm{B}(\mathrm{d}x_{[0:t]})||\mathbb{P}_\mathrm{A}(\mathrm{d}x_{[0:t]}))$, we apply Ito's formula to D_t^2 and get, under $\mathbb{P}_\mathrm{A}$

$$D_t^2 = 1+\sqrt{2}\int_0^t D_s^2\langle\Delta_s,\mathrm{d}W_s\rangle + \int_0^t \frac{D_s^2\|\Delta_s\|^2}{2}\mathrm{d}s. \tag{33}$$

Putting $\eta_t(x_t) = \|\Delta_t(x_t)\|^2$ and $\tilde{f}_t(x_t) = -x_t + 2v$, we have

$$D_t^2\eta_t = \eta_0 + \sqrt{2}\int_0^t D_s^2\langle\eta_s\Delta_s + \nabla\eta_s, \mathrm{d}W_s\rangle + \\ + \int_0^t D_s^2\left\{\eta_s\frac{\|\Delta_s\|^2}{2} + \nabla\eta_s(\tilde{f}(X_s)+2\Delta_s) + \frac{\partial\eta}{\partial s} + \mathrm{Tr}\big(\nabla^2\eta_s\big)\right\}\mathrm{d}s. \tag{34}$$

To study (33) and (34), we use (32) together with the following bounds which are consequences of the lemma's assumptions:

$$\begin{aligned}
\|\Delta_s(x_s)\| &\le c(M_1)(1+\|x_s\|+\|v\|)\\
\eta_s(x_s) &\le c(M_1)(1+\|x_s\|+\|v\|)^2\\
\|\nabla\eta_s(x_s)\| &\le c(M_1,M_2)(1+\|x_s\|+\|v\|)^{1+\alpha_2}\\
\left\|\frac{\partial\eta}{\partial s}(s,X_s)\right\| &\le c(M_1,\bar{M}_1)(1+\|x_s\|+\|v\|)^{1+\bar{\alpha}_1}\\
\big\|\mathrm{Tr}\big(\nabla^2\eta_s\big)\big\| &\le c(M_1,M_2,M_3)(1+\|x_s\|+\|v\|)^{1+\alpha_2+\alpha_3}.
\end{aligned}$$

The last inequality is justified by the fact that $\mathrm{Tr}\big(\nabla^2\eta_s\big) \lesssim \big\|\nabla^2\eta_s\big\|$, where by considering $\nabla^2\eta_s(x_s)$ as the second differential of η_s at x_s (i.e. a bilinear form from $\mathbb{R}^d\times\mathbb{R}^d$ to $\mathbb{R}$), we have

$$\frac{\partial^2\eta_s}{\partial x^2}(x)[h,k] = 2\left[\Delta_s(x)\circ\frac{\partial^2\Delta}{\partial x^2}(x)[h,k] + (\frac{\partial\Delta}{\partial x}(x)h)\circ(\frac{\partial\Delta}{\partial x}(x)k)\right], \forall(h,k)\in\mathbb{R}^d\times\mathbb{R}^d$$

where $\circ$ stands for the usual scalar product between two vectors in $\mathbb{R}^d$. These bounds show that the stochastic integrals (w.r.t. $\mathrm{d}W_s$) in (33) and (34) are true martingales (as opposed to merely local martingales). Moreover, there exist a constant $\widetilde{M}$ depending on M_1, M_2, M_3, $\bar{M}_1$, $\bar{\alpha}_1$, α_2, and α_3; and a constant $\widetilde{M}_1$ depending on M_1 only such that

$$\mathbb{E}\left[D_s^2\left|\eta_s\frac{\|\Delta_s\|^2}{2} + \nabla\eta_s(\tilde{f}(X_s)+2\Delta_s) + \frac{\partial\eta}{\partial s} + \mathrm{Tr}\big(\nabla^2\eta_s\big)\right|\right] \le 4\widetilde{M}e^{t\widetilde{M}_1(\|x_0\|^2+\|v\|^2)}\times \\ \times(1+\|x_0\|+\|v\|)^{4(1+\bar{\alpha}_1+\alpha_2+\alpha_3)}, \forall s\le t\le\frac{1}{\widetilde{M}_1}.$$

Taking expectation of both sides of (34) and rearranging yields

$$\big|\mathbb{E}[D_t^2\eta_t]-\eta_0\big| \le 4\mathbf{t}\widetilde{M}e^{t\widetilde{M}_1(\|x_0\|^2+\|v\|^2)}(1+\|x_0\|+\|v\|)^{4(1+\bar{\alpha}_1+\alpha_2+\alpha_3)}, \forall t\le\frac{1}{\widetilde{M}_1}.$$

Then we have, by taking expectation of both sides of (33):

$$\left|\chi^2(\mathbb{P}_{\mathrm{B}}(\mathrm{d}x_{[0:t]})||\mathbb{P}_{\mathrm{A}}(\mathrm{d}x_{[0:t]})) - t\frac{\eta_0}{2}\right| = \left|\mathbb{E}(D_t^2) - 1 - t\frac{\eta_0}{2}\right| = \left|\int_0^t \left(\mathbb{E}\left[\frac{D_s^2\eta_s}{2}\right] - \frac{\eta_0}{2}\right)\mathrm{d}s\right| \leq \int_0^t \left|\mathbb{E}\left[\frac{D_s^2\eta_s}{2}\right] - \frac{\eta_0}{2}\right|\mathrm{d}s$$
$$\leq \int_0^t 2s\widetilde{M}e^{t\widetilde{M}_1(\|x_0\|^2+\|v\|^2)}(1+\|x_0\|+\|v\|)^{4(1+\bar{\alpha}_1+\alpha_2+\alpha_3)}\mathrm{d}s$$
$$= t^2\widetilde{M}e^{t\widetilde{M}_1(\|x_0\|^2+\|v\|^2)}(1+\|x_0\|+\|v\|)^{4(1+\bar{\alpha}_1+\alpha_2+\alpha_3)}.$$

The proof is completed. □

We are now ready to give the proof of Proposition 3.3.

Proof. To make the arguments clearer, we shall assume that the proposal distribution is

$$\hat{\pi}(x_k|x_{k+1}) = \mathcal{N}(x_k|\sqrt{1-\alpha_{k+1}}x_{k+1} + 2(1-\sqrt{1-\alpha_{k+1}})\nabla\log\hat{g}_{k+1}(x_{k+1}), \alpha_{k+1}I)$$

which is slightly different from (16). As we will see, the proof also applies to the original discretization with minimal changes. For $0 < s < u < T$, the distributions $\hat{\pi}(x_s|x_u)$ and $\pi(x_s|x_u)$ can be obtained by respectively solving the following SDEs between times $T-u$ and $T-s$:

$$\hat{\pi}(x_s|x_u): \quad \mathrm{d}Y_t^{\mathrm{A}} = (-Y_t^{\mathrm{A}} + 2\nabla\log\hat{g}_{T-\mathbf{u}}(Y_t^{\mathrm{A}}))\mathrm{d}t + \sqrt{2}\mathrm{d}W_t^{\mathrm{A}}, \quad Y_{T-u}^{\mathrm{A}} = x_u$$
$$\pi(x_s|x_u): \quad \mathrm{d}Y_t^{\mathrm{B}} = (-Y_t^{\mathrm{B}} + 2\nabla\log g_{T-t}(Y_t^{\mathrm{B}}))\mathrm{d}t + \sqrt{2}\mathrm{d}W_t^{\mathrm{B}}, \quad Y_{T-u}^{\mathrm{B}} = x_u.$$

The assumptions in Section C.4.1 and Lemma C.7 show that the conditions of Lemma C.8 are satisfied for this pair of SDEs. Thus

$$\left|\chi^2(\mathbb{P}_{\mathrm{B}}(\mathrm{d}y_{[T-u,T-s]})||\mathbb{P}_{\mathrm{A}}(\mathrm{d}y_{[T-u,T-s]})) - 2(u-s)\|\nabla\log g_u(x_u) - \nabla\log\hat{g}_u(x_u)\|^2\right| \leq \widetilde{M}(u-s)^2\times$$
$$\times\, e^{(u-s)\widetilde{M}_1(\|x_u\|^2+\|\nabla\log\hat{g}_u(x_u)\|^2)}(1+\|x_u\|+\|\nabla\log\hat{g}_u(x_u)\|)^{\alpha_+}$$

for $\alpha_+ = 4(1+\bar{\alpha}_1+\alpha_2+\alpha_3)$ and $u - s \leq \frac{1}{\widetilde{M}_1}$. This, together with the data processing inequality and the assumption on $|\nabla\log\hat{g}_t|$, implies

$$\chi^2(\pi(x_k|x_{k+1})||\hat{\pi}(x_k|x_{k+1})) \leq 2\delta\|\nabla\log g_{k+1} - \nabla\log\hat{g}_{k+1}\|^2(x_{k+1}) + \widetilde{M}\delta^2e^{\delta\widetilde{M}_1(1+2C_2^2)(1+\|x_{k+1}\|)^2}\times$$
$$\times\,(1+C_2)^{\alpha_+}(1+\|x_{k+1}\|)^{\alpha_+}, \forall\delta \leq \frac{1}{\widetilde{M}_1}.$$

Thus, for a sufficiently fine discretization,

$$\sum_k \int \frac{\pi_{k+1}(x_{k+1})^2}{\hat{\pi}_{k+1}(x_{k+1})}\chi^2(\pi(x_k|x_{k+1})||\hat{\pi}(x_k|x_{k+1}))\mathrm{d}x_{k+1} \leq$$
$$\leq \sum_k \delta\int\frac{\pi_{k+1}(x_{k+1})^2}{\hat{\pi}_{k+1}(x_{k+1})}2\|\nabla\log g_{k+1} - \nabla\log\hat{g}_{k+1}\|^2(x_{k+1})\mathrm{d}x_{k+1}+$$
$$+\sum_k\int C_1\pi_{k+1}(x_{k+1})\widetilde{M}\delta^2e^{\delta\widetilde{M}_1(1+2C_2^2)(1+\|x_{k+1}\|^2)}(1+C_2)^{\alpha_+}(1+\|x_{k+1}\|)^{\alpha_+}\mathrm{d}x_{k+1}. \quad (35)$$

The first term is a Riemann sum and converges to $\int_0^T\int\frac{\pi_t(x_t)^2}{\hat{\pi}_t(x_t)}2\|\nabla\log g_t - \nabla\log\hat{g}_t\|^2(x_t)\mathrm{d}x_t\mathrm{d}t$. To bound the second term, first note that

$$\mathbb{E}_{\pi_{k+1}}[(1+\|X\|)^{2\alpha_+}] \leq 2^{2\alpha_+-1}\mathbb{E}\left[1+\|X\|^{2\alpha_+}\right] \leq 2^{2\alpha_+-1}\mathbb{E}\left[1+e^{\vartheta\|X\|^2}\max\left(\frac{\lceil\alpha_+\rceil!}{\vartheta^{\lceil\alpha_+\rceil}}, \frac{\lfloor\alpha_+-1\rfloor!}{\vartheta^{\lfloor\alpha_+-1\rfloor}}\right)\right]$$
$$\leq 2^{2\alpha_+-1}\left(1+M_\infty\max\left(\frac{\lceil\alpha_+\rceil!}{\vartheta^{\lceil\alpha_+\rceil}}, \frac{\lfloor\alpha_+-1\rfloor!}{\vartheta^{\lfloor\alpha_+-1\rfloor}}\right)\right)$$

where M_θ and ϑ appear in Assumption C.5. Thus, as long as $\delta \widetilde{M}_1(1+2C_2^2) \leq \vartheta/2$, it holds that

$$\int \pi_{k+1}(x) e^{\delta \widetilde{M}_1(1+2C_2^2)\|x\|^2}(1+\|x\|)^{\alpha_+}\mathrm{d}x \leq \mathbb{E}_{\pi_{k+1}}\left[e^{\vartheta\|X\|^2/2}(1+\|X\|)^{\alpha_+}\right]$$
$$\leq \mathbb{E}^{1/2}\left[e^{\vartheta\|X\|^2}\right]\mathbb{E}^{1/2}\left[(1+\|X\|)^{2\alpha_+}\right] \leq C(M_\infty, \alpha_+, \vartheta)$$

for some constant $C(M_\infty, \alpha_+)$ depending on M_∞, α_+, and ϑ. Hence the second sum of (35) tends to 0 when $\delta \to 0$. The proof is finished. □

C.5. Proof of Proposition 4.1

The denoising score matching identity is standard and recalled here for convenience. We have

$$\pi_k(x_k) = \int p(x_k|x_0)\pi_0(x_0)\mathrm{d}x_0$$

so by using the log derivative we obtain

$$\nabla \log \pi_k(x_k) = \int \nabla \log p(x_k|x_0)\frac{\pi_0(x_0)p(x_k|x_0)}{\pi_k(x_k)}\mathrm{d}x_0 \tag{36}$$
$$= \int \nabla \log p(x_k|x_0)\pi(x_0|x_k)\mathrm{d}x_0. \tag{37}$$

It can be easily verified that the interchange of differentiation and integration here does not require any regularity assumption on $\pi_0(x_0)$ apart from differentiability.

To prove the novel score identity, we first note that, under the condition $\int \|\nabla\pi(x_0)\| e^{-\eta\|x_0\|^2}\mathrm{d}x_0 < \infty, \forall \eta > 0$, we have

$$\int \nabla_{x_0} \log \pi(x_0|x_k)\pi(x_0|x_k)\mathrm{d}x_0 = 0 \tag{38}$$

according to Lemma C.9. Combining this identity with (37), we have, for any $\alpha \in \mathbb{R}$,

$$\nabla \log \pi(x_k) = \int \left[\nabla_{x_k} \log p(x_k|x_0) + \alpha \nabla_{x_0} \log \pi(x_0|x_k)\right]\pi(x_0|x_k)\mathrm{d}x_0.$$

In particular:

- For $\alpha = \frac{1}{\sqrt{1-\lambda_k}}$, we get
$$\nabla \log \pi(x_k) = \frac{1}{\sqrt{1-\lambda_k}}\int \nabla \log \pi(x_0)\pi(x_0|x_k)\mathrm{d}x_0$$
which is the identity presented in Appendix C.1.3 (De Bortoli et al., 2021);
- For $\alpha = \sqrt{1-\lambda_k}$, we get
$$\nabla \log \pi(x_k) = \int \left(\sqrt{1-\lambda_k}\nabla \log g_0(x_0) - x_k\right)\pi(x_0|x_k)\mathrm{d}x_0$$
which is the identity we wanted to prove.

The verifications are straightforward by remarking that $\nabla_{x_0} \log \pi(x_0|x_k) = \nabla \log g_0(x_0) + \nabla_{x_0} \log p(x_0|x_k)$. We also note that choosing $\alpha = 0$ brings us back to the classical score matching loss. Therefore, different values of α give losses with different properties.

We finish this section with a technical lemma giving conditions for (38) to hold. The identity is a particular case of what is known in the literature as zero-variance control variates and Stein's control variates (Assaraf & Caffarel, 1999; Mira et al., 2013; Anastasiou et al., 2023).

Lemma C.9. *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a probability density, i.e. $f \geq 0$ and $\int f(x)\mathrm{d}x = 1$. Suppose that f is continuously differentiable and $\int \|\nabla f(x)\|\mathrm{d}x < \infty$. Then $\int \nabla f(x)\mathrm{d}x = 0$.*

Remark C.10. The condition $\int \|\nabla f(x)\|\mathrm{d}x < \infty$ is clearly the minimum necessary for $\int \nabla f(x)\mathrm{d}x = 0$ to make sense. On the other hand, we do *not* explicitly require that f or ∇f vanishes at infinity.

Proof. Without loss of generality, we only prove that $\int \partial_1 f(x)\mathrm{d}x = 0$. Put $g(x_1) := \int f(x_1, x_{2:d})\mathrm{d}x_{2:d}$. Fubini's theorem then implies that $\int_{\mathbb{R}} g(x_1)\mathrm{d}x_1 = 1$. We have

$$\begin{aligned}\int_{\mathbb{R}^d} \partial_1 f(x)\mathrm{d}x &= \lim_{M\to\infty} \int_{\mathbb{R}^{d-1}} \int_{-M}^{M} \partial_1 f(x_1, x_{2:d})\mathrm{d}x_1\mathrm{d}x_{2:d} = \lim_{M\to\infty} \int_{\mathbb{R}^{d-1}} f(M, x_{2:d}) - f(-M, x_{2:d})\mathrm{d}x_{2:d} \\ &= \lim_{M\to\infty} g(M) - g(-M).\end{aligned} \tag{39}$$

Put $I(M) := \int_0^\infty |g(M+x) - g(-M-x)|\mathrm{d}x$. We have

$$I(M) \leq \int_0^\infty |g(M+x)| + |g(-M-x)|\mathrm{d}x,$$

thus $\lim_{M\to\infty} I(M) = 0$ by the integrability of g. Combining this with Fatou's lemma yields

$$0 = \liminf_{M\to\infty} I(M) \geq \int_0^\infty \liminf_{M\to\infty} |g(M+x) - g(-M-x)|\mathrm{d}x = \int_0^\infty \left|\int_{\mathbb{R}^d} \partial_1 f(y)\mathrm{d}y\right|\mathrm{d}x$$

by (39). This means that $\int_{\mathbb{R}^d} \partial_1 f(y)\mathrm{d}y = 0$. □

C.6. Proof of Proposition 4.2

Proof. Since we are in the Gaussian case with $d = 1$, we have $\nabla \log g_0(x_0) = ax_0 + b$ for some $a, b \in \mathbb{R}$. Therefore $\ell_{\text{NSM}}(\theta)$ and $\mathrm{Var}(\hat{\ell}_{\text{DSM}}(\theta))$ are trivially bounded. To study $\ell_{\text{DSM}}(\theta)$, we first note that

$$\mathrm{Var}(X_0|X_t) = \frac{\lambda_t\sigma^2}{\lambda_t + (1-\lambda_t)\sigma^2} =: \rho_t.$$

Write

$$\begin{aligned}\mathbb{E}\left[\|s_\theta(t, X_t) - \nabla \log p(X_t|X_0)\|^2 \middle| X_t\right] &\geq \mathrm{Var}\left(s_\theta(t, X_t) - \nabla \log p(X_t|X_0) \middle| X_t\right) = \mathrm{Var}\left(-\frac{X_t - \sqrt{1-\lambda_t}X_0}{\lambda_t} \middle| X_t\right) \\ &= \frac{1-\lambda_t}{\lambda_t^2}\rho_t\end{aligned}$$

so

$$\ell_{\text{DSM}}(\theta) \geq \int_0^T \frac{1-\lambda_t}{\lambda_t^2}\rho_t\mathrm{d}t = \infty$$

since $\rho_t \sim 2t$ as $t \to 0$. Concerning $\hat{\nabla}\ell_{\text{DSM}}(\theta) = 2T(s_\theta(\tau, X_\tau) - \nabla \log p(X_\tau|X_0))\nabla_\theta s_\theta(\tau, X_\tau)$, we have

$$\begin{aligned}\mathbb{E}\left[\left\|\hat{\nabla}\ell_{\text{DSM}}(\theta)\right\|^2 \middle| X_t, \tau = t\right] &= 4T^2\|\nabla_\theta s_\theta(t, X_t)\|^2 \mathbb{E}\left[\|s_\theta(t, X_t) - \nabla \log p(X_t|X_0)\|^2 \middle| X_t, \tau = t\right] \\ &\geq 4T^2\|\nabla_\theta s_\theta(t, X_t)\|^2 \frac{1-\lambda_t}{\lambda_t^2}\rho_t\end{aligned}$$

so

$$\begin{aligned}\mathbb{E}\left[\left\|\hat{\nabla}\ell_{\text{DSM}}(\theta)\right\|^2\right] &= \frac{1}{T}\int_0^T \mathbb{E}\left[\left\|\hat{\nabla}\ell_{\text{DSM}}(\theta)\right\|^2 \middle| \tau = t\right]\mathrm{d}t \geq 4T\int_0^T \mathbb{E}\left[\|\nabla_\theta s_\theta(t, X_t)\|^2 \frac{1-\lambda_t}{\lambda_t^2}\rho_t\right]\mathrm{d}t \\ &= 4T\mathbb{E}\left[\int_0^T \|\nabla_\theta s_\theta(t, X_t)\|^2 \frac{1-\lambda_t}{\lambda_t^2}\rho_t\mathrm{d}t\right].\end{aligned}$$

The integral inside the last expectation is infinite whenever the event $\|\nabla_\theta s_\theta(0, X_0)\| \neq 0$ holds, thanks to the continuity of $\nabla_\theta s$ and the path $X_{[0,T]}$. Since $\mathbb{E}\|\nabla_\theta s_\theta(t, X_0)\|^2 > 0$ by assumption, that event has non-zero probability, which concludes the proof. □

D. Experimental Details

In this section we give additional details and ablations relating to our experimental results. We begin by providing details of the sampling tasks we considered. We then provide details of our implementation and the baseline methods. We finally provide additional ablation studies and results which demonstrate the properties of our method.

D.1. Benchmarking targets

Gaussian Here we consider the target $\pi(x) = \mathcal{N}(x; 2.75, 0.25^2)$.

Mixture This target was used in Arbel et al. (2021). It is an equally weighted mixture of 6 bivariate Gaussian distributions with means $\mu_1 = (3.0, 0.0), \mu_2 = (-2.5, 0.0), \mu_3 = (2.0, 3.0), \mu_4 = (0.0, 3.0), \mu_5 = (0.0, -2.5), \mu_6 = (3.0, 2.0)$ and covariances $\Sigma_1 = \Sigma_2 = \left(\begin{smallmatrix} 0.7 & 0.0 \\ 0.0 & 0.05 \end{smallmatrix}\right), \Sigma_4 = \Sigma_5 = \left(\begin{smallmatrix} 0.05 & 0.0 \\ 0.0 & 0.07 \end{smallmatrix}\right), \Sigma_3 = \Sigma_6 = \left(\begin{smallmatrix} 1.0 & 0.95 \\ 0.95 & 1.0 \end{smallmatrix}\right)$. This target is symmetric around $y = x$.

Funnel This target was proposed by Neal (2003). Its density follows $x_0 \sim \mathcal{N}(0, \sigma_f^2), x_{1:9}|x_0 \sim \mathcal{N}(0, \exp(x_0)\mathbf{I})$, with $\sigma_f = 3$.

Logistic Regression The Bayesian logistic regression model is defined by the prior distribution $\theta \sim \mathcal{N}(0, \sigma^2 \mathrm{I})$ and likelihood $y|\theta, x \sim \text{Bernoulli}(\sigma(\theta^T x))$ where σ is the sigmoid function. We consider sampling the posterior $\theta|y, x$ on the Ionosphere and Sonar datasets, which are of 35 and 61 dimensions respectively.

Brownian Motion In this task, we make noisy observations of a simple Brownian motion over 30 time steps. The model was introduced by Sountsov et al. (2020) and is defined by the prior $\alpha_{\text{inn}} \sim \text{LogNormal}(0, 2)$, $\alpha_{\text{obs}} \sim \text{LogNormal}(0, 2)$, $x_1 \sim \mathcal{N}(0, \alpha_{\text{inn}}^2)$ and $x_i \sim \mathcal{N}(x_{i-1}, \alpha_{\text{inn}}^2)$ for $i = 2, ..., 30$. The observation likelihood is given by $y_i \sim \mathcal{N}(x_i, \alpha_{\text{obs}}^2)$ for $i = 1, ..., 30$. The goal is to sample the posterior distribution of $\alpha_{\text{inn}}, \alpha_{\text{obs}}, x_1, ..., x_{30} | \{y_i\}_{i=1}^{10} \bigcup \{y_i\}_{i=21}^{30}$. This task is in 32 dimensions.

Log Gaussian Cox Process The LGCP model (Møller et al., 1998) was developed for the analysis of spatial data. The Poisson rate parameter $\lambda(x)$ is modelled on the grid using an exponentially-transformed Gaussian process, and observations come from a Poisson point process. The unnormalized posterior density is given directly by $\gamma(x) = \mathcal{N}(x; \mu, K) \prod_{i \in [1:M]^2} \exp(x_i y_i - a \exp(x_i))$, where x_i are the points of a regular $M \times M$ grid. In our experiments, we fit this model on the Pines forest dataset where $M = 40$, resulting in a problem in 1600 dimensions.

GMM The challenging Gaussian Mixture Model used in (Midgley et al., 2023) is an unequally weighted mixture of 40 Gaussian components. The mean of each component is uniformly distributed in the range $[-40, 40]^d$, the covariance is $\sigma^2 I_d$ where $\sigma = \log(1 + \exp(0.1))$ and the unnormalized weight is uniformly distributed in $[0, 1]$. We consider dimensions $d \in \{1, 2, 5, 10, 20\}$.

D.2. Algorithmic details and hyperparameter settings

Here we give details of the algorithmic settings used in our experiments. We first describe the considerations taken to ensure a fair comparison between baselines, and then we detail exact hyperparameter settings.

Our method was implemented in Python using the libraries of JAX (Bradbury et al., 2018), Haiku and Optax. Our implementation is available on Github[1]. We used the open source code-bases of Arbel et al. (2021) to run the SMC and CRAFT baselines and of Vargas et al. (2023) to run the PIS and DDS benchmarks, both of which are also implemented in JAX.

In all experiments we used 2000 particles to estimate the normalizing constant.

Variational approximation We used a variational approximation as the reference distribution for all methods. We found that this was required for numerical stability of the potential function $g_t(x_t)$ in our method. We therefore used the same variational approximation for all methods to ensure a fair comparison. Note that PIS reverses a pinned brownian motion

[1] https://github.com/angusphillips/particle_denoising_diffusion_sampler

	Gaussian (16)	Mixture (16)	Funnel (32)	Brownian (16)	Ion (32)	Sonar (32)	LGCP (128)
PDDS	37570 / 84 / 0.10	37764 / 84 / 0.13	39316 / 114 / 0.15	43584 / 90 / 0.19	44166 / 97 / 0.12	49210 / 105 / 0.12	347776 / 2492 / 1.86
CRAFT	32 / 4 / 0.02	176608 / 26 / 0.09	6077440 / 178 / 0.18	1024 / 27 / 0.21	2240 / 38 / 0.09	3904 / 40 / 0.09	409600 / 3072 / 14.9
PIS	37570 / 129 / 0.00	37764 / 167 / 0.01	39316 / 394 / 0.02	43854 / 176 / 0.01	44166 / 324 / 0.02	49210 / 326 / 0.01	347776 / 3931 / 0.64
DDS	37570 / 136 / 0.00	37764 / 187 / 0.01	39316 / 381 / 0.02	43854 / 183 / 0.01	44166 / 338 / 0.01	49210 / 332 / 0.02	347776 / 3941 / 0.68
SMC	0 / 0 / 0.02	0 / 0 / 0.07	0 / 0 / 0.17	0 / 0 / 0.20	0 / 0 / 0.09	0 / 0 / 0.09	0 / 0 / 14.6

Table 1: Number of trainable parameters / training time total (seconds) / sampling time per 2000 particles (seconds). Timings are averaged over 3 training seeds.

and thus the reference distribution depends on the diffusion time span T and the noise coefficient σ. Since σ affects the performance of the PIS algorithm itself, we tune this parameter independently rather than setting this via the variational approximation. The variational approximation was a mean-field variational distribution i.e. a diagonal Gaussian distribution learnt by optimizing the ELBO. We used $20,000$ optimisation steps ($50,000$ for the `Funnel` and `Brownian` tasks) with the Adam optimizer (Kingma & Ba, 2015) and learning rate $1e-3$. We did not use a variational approximation in the `Gaussian` and `GMM` tasks where we used $\mathcal{N}(0,1)$ and $\mathcal{N}(0,20^2 I_d)$ respectively.

Network architectures and optimizer settings For the CRAFT baseline, we followed the flow network architectures and optimizer settings given in Matthews et al. (2022), which are restated below for completeness. For the diffusion-based methods (PDDS, DDS and PIS) we use the same network architecture and optimizer settings for each method. The neural network follows the PISGRAD network of Zhang & Chen (2022) with minor adaptations. We use a sinusoidal embedding of 128 dimensions for the time input. We use a 3-layer MLP with 64 hidden units per layer for the 'smoothing' network ($r_\eta(t)$ in our notation and $\mathrm{NN}_2(t)$ in Zhang & Chen (2022)). For the main potential/score network (N_γ in our notation and $\mathrm{NN}_1(t,x)$ in Zhang & Chen (2022)), we use a 2 layer MLP of 64 hidden units per layer to encode the 128-dimensional time embedding. This is concatenated with the state input x before passing through a 3-layer MLP with 64 hidden units per layer, outputting a vector of dimension d. In PDDS, we take the scalar product of this output with the state input x to approximate the potential function, while PIS and DDS use the d-dimensional output to approximate the optimal control term. The activation function is GeLU throughout. We train for $10,000$ iterations of the Adam optimizer (Kingma & Ba, 2015) with batch size 300 and a learning rate of $1e-3$, which decays exponentially at a rate of 0.95 every 50 iterations (with the exception of the `Funnel` task where we did not use any learning rate decay).

The number of trainable parameters for each method and task can be found in Table 1, along with training time and sampling time (performed on a NVIDIA GeForce GTX 1080 Ti GPU).

Annealing and noise schedules For the annealing based approaches (SMC and CRAFT) we used a geometric annealing schedule with initial distribution the variational approximation as described above. For the diffusion based approaches (PDDS, DDS and PIS) we carefully considered the appropriate noise schedules for each method. Firstly, we fix the diffusion time span at $T=1$ and adapt the discretization step size depending on the number of steps K of the experiment, i.e. $\delta = T/K$. This choice is equivalent to the fixed discretization step and varying diffusion time T as considered by Vargas et al. (2023), up to the choice of α_{max}.

For PIS, the original work of Zhang & Chen (2022) used by default a uniform noise schedule controlled by σ. Further, Vargas et al. (2023) were unable to find a noise schedule which improved performance above the default uniform noise schedule. As such as we stick with the uniform noise schedule and tune the noise coefficient σ by optimizing the ELBO objective over a grid search.

For DDS, it was observed by Vargas et al. (2023) that controlling the transition scale $\sqrt{\alpha_k}$ such that it goes smoothly to zero as $k \to 0$ is critical to performance. To achieve this they choose a cosine-based schedule $\alpha_k^{1/2} = \alpha_{\text{max}}^{1/2} \cos^2\left(\frac{\pi}{2}\frac{1-k/K+s}{1+s}\right)$ for s small (0.008 following Nichol & Dhariwal (2021)), which we term the DDS cosine schedule. The parameter α_{max} is tuned such that the noise at the final step in the reverse process is sufficiently small. We found that this scheduler did indeed result in the best performance when compared to a linear noise schedule ($\beta_t = \beta_0 + \beta_T t/T$) or the popular cosine schedule of Nichol & Dhariwal (2021) ($\lambda_t = 1 - \cos^2\left(\frac{\pi}{2}\frac{t/T+s}{1+s}\right)$). As such we use the DDS cosine schedule and tune α_{max} by optimizing the ELBO objective over a grid search.

For our method PDDS, we obtained the best performance using the cosine schedule of Nichol & Dhariwal (2021), which sets $\lambda_t = 1 - \cos^2\left(\frac{\pi}{2}\frac{t/T+s}{1+s}\right)$ where we recall that $\lambda_t = 1 - \exp\left(-2\int_0^t \beta_s ds\right)$, i.e. the variance of the transition from 0 to t. We provide an illustration of the benefits of this schedule in the following section. In particular we found that the alternative DDS cosine schedule did not improve performance and added the complexity of tuning the parameter α_{max}.

In summary, while each of the diffusion based approaches used different noise schedulers, each was chosen to provide the best performance for the individual approach and thus ensures a fair comparison.

D.2.1. SMC and CRAFT settings

We used 1 iteration of an HMC kernel with 10 leapfrog integrator steps as the proposal distribution in the SMC and CRAFT baselines. We tuned the HMC step sizes based on initial runs and obtained the step size schedules given below. We performed simple resampling when the ESS dropped below 30%. We trained CRAFT for 500 iterations (1000 on `Funnel`) with a batch size of 2000 and learning rate schedule detailed below. We also list the flow architecture in each task. Our parameter settings differ to those in Matthews et al. (2022) since we use the variational approximation, which results in larger MCMC step sizes and smaller learning rates.

Gaussian Step sizes $[0.7, 0.7, 0.5, 0.4]$ linearly interpolated between times $[0.0, 0.25, 0.5, 1.0]$. CRAFT used a diagonal affine flow, with a learning rate of $1e-2$.

Mixture Step sizes $[0.5, 0.5, 0.5, 0.3]$ linearly interpolated between times $[0.0, 0.25, 0.5, 1.0]$. CRAFT used a spline inverse autoregressive flow with 10 spline bins and a 3 layer autoregressive MLP of 30 hidden units per layer, with a learning rate of $1e-3$.

Funnel Step sizes $[1.0, 0.9, 0.8, 0.7, 0.6]$ linearly interpolated between times $[0.0, 0.25, 0.5, 0.75, 1.0]$. CRAFT used an affine inverse autoregressive flow, trained for 4000 iterations with a learning rate of $1e-3$.

Brownian Step sizes $[0.8, 0.8, 0.7, 0.6, 0.5]$ linearly interpolated between times $[0.0, 0.25, 0.5, 0.75, 1.0]$. CRAFT used a diagonal affine flow, with a learning rate of $1e-3$.

Ion Step sizes $[0.7, 0.7, 0.6, 0.5, 0.4]$ linearly interpolated between times $[0.0, 0.1, 0.25, 0.5, 1.0]$. CRAFT used a diagonal affine flow, with a learning rate of $1e-3$.

Sonar Step sizes $[0.7, 0.7, 0.6, 0.5, 0.35]$ linearly interpolated between times $[0.0, 0.1, 0.25, 0.5, 1.0]$. CRAFT used a diagonal affine flow, with a learning rate of $1e-3$.

LGCP Step sizes $[0.35, 0.35, 0.3, 0.2]$ linearly interpolated between times $[0.0, 0.25, 0.5, 1.0]$. CRAFT used a diagonal affine flow, with a learning rate of $1e-4$.

GMM1 Step sizes $[5, 4, 3, 2.8, 2.5]$ linearly interpolated between times $[0.0, 0.3, 0.5, 0.85, 1.0]$. CRAFT used a diagonal affine flow, with a learning rate of $1e-3$.

GMM2 Step sizes $[4, 3, 2.5, 2.1, 2]$ linearly interpolated between times $[0.0, 0.3, 0.5, 0.85, 1.0]$. CRAFT used a diagonal affine flow, with a learning rate of $1e-3$.

GMM5 Step sizes $[5, 3.3, 2.3, 1.8, 1.6]$ linearly interpolated between times $[0.0, 0.25, 0.5, 0.8, 1.0]$. CRAFT used a diagonal affine flow, with a learning rate of $1e-3$.

	Base step	1	2	4	8	16
Gaussian	1	0.86	0.86	1.00	0.96	0.82
Mixture	1	0.28	0.28	0.36	0.52	0.54
Brownian	1	0.76	0.76	0.84	0.80	0.72
Funnel	2	0.60	0.68	0.68	0.60	0.64
Ion	2	0.68	0.80	0.74	0.64	0.52
Sonar	2	0.68	0.82	0.78	0.64	0.50
LGCP	8	0.74	0.62	0.60	0.44	0.26
GMM1	2	0.22	0.28	0.24	0.20	0.22
GMM2	2	0.14	0.18	0.18	0.16	0.16
GMM5	2	0.20	0.28	0.26	0.24	0.20
GMM10	4	0.36	0.34	0.30	0.26	0.22
GMM20	8	0.40	0.32	0.32	0.26	0.18

Table 2: Optimal settings for α_{max}. The number of steps for a given entry is the base steps in the first column multiplied by the step multiplier in the zeroth row.

	Steps	1	2	4	8	16
Gaussian	1	NA	1.00	1.00	1.00	1.00
Mixture	1	NA	2.40	2.20	1.92	1.88
Brownian	1	NA	0.08	0.10	0.10	0.13
Funnel	2	1.50	1.00	1.00	1.00	1.00
Ion	2	0.37	0.40	0.40	0.46	0.46
Sonar	2	0.25	0.31	0.40	0.46	0.49
LGCP	8	1.36	1.64	1.78	1.99	2.06
GMM1	2	15.00	14.00	15.00	15.00	16.00
GMM2	2	4.40	1.30	1.30	7.30	8.70
GMM5	2	1.30	1.30	1.40	1.40	1.40
GMM10	4	1.30	1.30	1.30	1.30	1.30
GMM20	8	1.30	1.30	1.30	1.20	1.20

Table 3: Optimal settings for σ. The number of steps for a given entry is the base steps in the first column multiplied by the step multiplier in the zeroth row. We were unable to tune PIS with one step size.

GMM10 Step sizes $[3, 2, 1.5, 1.5]$ linearly interpolated between times $[0.0, 0.5, 0.85, 1.0]$. CRAFT used a diagonal affine flow, with a learning rate of $1e-3$.

GMM20 Step sizes $[3, 1.8, 1.4, 1.3]$ linearly interpolated between times $[0.0, 0.5, 0.85, 1.0]$. CRAFT used a diagonal affine flow, with a learning rate of $1e-3$.

We used SMC with the above settings for 1000 steps to estimate the ‘ground truth’ normalizing constant on the Bayesian posterior targets.

D.2.2. DDS SETTINGS

Optimal settings for α_{max} are given in Table 2. Note that we do not tune σ as in Vargas et al. (2023) since we used a variational approximation. We also tuned DDS on the GMM tasks but did not present the results as they were not competitive with PDDS and CRAFT.

D.2.3. PIS SETTINGS

Optimal settings for σ are given in Table 3. Note that we were unable to obtain reasonable performance with PIS with only 1 step. We also tuned PIS on the GMM tasks but did not present the results as they were not competitive with PDDS and CRAFT.

D.2.4. PDDS settings

No tuning of the cosine noise schedule was required. We performed systematic resampling (Douc & Cappé, 2005) when the ESS dropped below 30%. PDDS-MCMC used 10 Metropolis-adjusted Langevin MCMC steps with step sizes tuned based on initial runs with the initial simple approximation, targeting an acceptance rate of approximately 0.6. The resulting step sizes can be found below. We used 20 iterations of PDDS, each trained for 500 steps with a fresh instance of the learning rate schedule ($1e-3$ with exponential decay at a rate of 0.95 per 50 iterations). At each refinement we initialise the potential approximation at it's previous state, rather than training from scratch at each refinement. We found that for a limited computational budget, better performance was obtained for a fast iteration rate (20 iterations with 500 training steps each). We also tested a slower iteration rate (2 iterations with 10,000 training steps each) which performed equivalently but required a larger overall training budget. The fast iteration schedule uses a lower number of density evaluations but has a larger training time due to requiring more frequent compilation of the PDDS sampler.

Gaussian Step sizes $[0.1, 0.2, 0.5, 0.6]$ linearly interpolated between times $[0, 0.5, 0.75, 1.0]$.

Mixture Step sizes $[0.05, 0.15, 0.4, 0.6]$ linearly interpolated between times $[0, 0.5, 0.75, 1.0]$.

Funnel Step sizes $[0.4, 0.3, 0.5, 0.6, 0.6]$ linearly interpolated between times $[0, 0.2, 0.5, 0.75, 1.0]$. We found that the Langevin MCMC can become unstable on the `Funnel` task due to extreme gradients of the density function, therefore at each iteration of PDDS we reduced the step sizes by 50% for the first 10 iterations.

Brownian Step sizes $[0.2, 0.4, 0.5, 0.5]$ linearly interpolated between times $[0, 0.5, 0.75, 1.0]$.

Ion Step sizes $[0.15, 0.25, 0.5, 0.6]$ linearly interpolated between times $[0, 0.5, 0.75, 1.0]$.

Sonar Step sizes $[0.1, 0.1, 0.18, 0.32, 0.4]$ linearly interpolated between times $[0, 0.25, 0.5, 0.75, 1.0]$.

LGCP Step sizes $[0.1, 0.1, 0.15, 0.2]$ linearly interpolated between times $[0, 0.5, 0.75, 1.0]$.

GMM1 Step sizes $[10, 11, 20, 35, 100]$ linearly interpolated between times $[0.0, 0.25, 0.5, 0.75, 1.0]$.

GMM2 Step sizes $[2, 2.5, 4, 6, 12, 50]$ linearly interpolated between times $[0.0, 0.25, 0.5, 0.6, 0.75, 1.0]$.

GMM5 Step sizes $[1.5, 1.5, 2.5, 4, 8, 30]$ linearly interpolated between times $[0.0, 0.25, 0.5, 0.6, 0.75, 1.0]$.

GMM10 Step sizes $[1, 1.2, 1.8, 3, 6, 20]$ linearly interpolated between times $[0.0, 0.25, 0.5, 0.6, 0.75, 1.0]$.

GMM20 Step sizes $[0.8, 1.0, 2, 3, 6, 20]$ linearly interpolated between times $[0.0, 0.25, 0.5, 0.6, 0.75, 1.0]$.

D.3. Ablation studies

Importance of SMC and iterative potential approximations Here we study the behaviour of PDDS on a gaussian mixture task where the initial (here referred to as 'naive') approximation Equation (8) only captures one out of three modes when simulating the reverse SDE Equation (7). Figure 8 shows that our method is able to recover the unknown modes after only two iterations of potential training. Following the discussion in Section 4.5, there are two mechanisms at play which allow our method to recover additional modes which are missed by the naive approximation.

Firstly, our asymptotically correct SMC scheme Algorithm 1 means that we do not simply re-learn the potential function of the previous iteration during training, but instead we learn an improved potential function since the training samples are 'improved' by the SMC scheme. We evidence the improvement in potential functions by plotting the path of distributions induced by Equation (7) with the learnt potential at each PDDS iteration in Figure 9. We see that at each iteration of PDDS the sequence of distributions moves closer to the true denoising sequence for the given target. Furthermore we show that our SMC scheme is critical by showing the behaviour of PDDS if the SMC scheme is ignored (i.e. remove all resampling steps). Figure 10 and Figure 11 show the resulting samples and sequence of distributions when we simply simulate the reverse SDE Equation (7) using the previous potential approximation without applying the SMC correction scheme. We observe that very

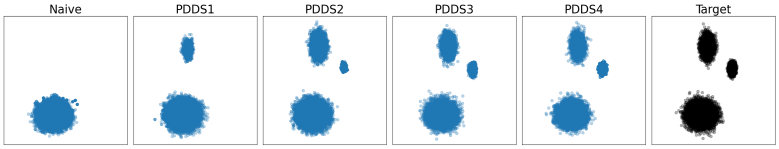

Figure 8: Samples from PDDS on a 3-mode mixture of Gaussians.

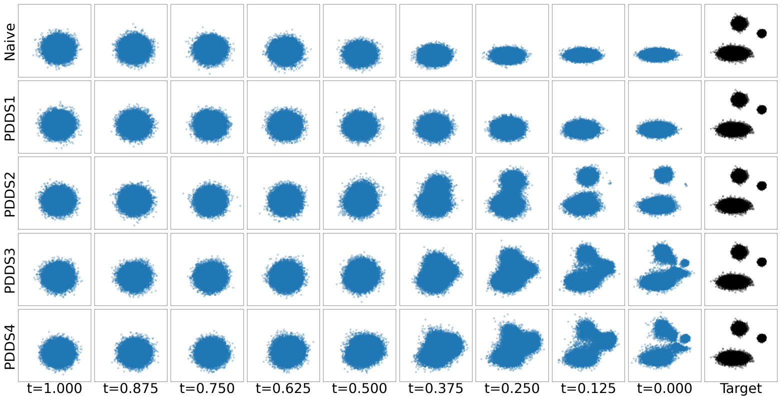

Figure 9: Left to right: marginal distributions of the reverse SDE Equation (7) using the potential approximation at each iteration of PDDS.

few particles do still reach the missing modes but this is not compounded in each iteration and after the first iteration no improvements are made.

The second, more subtle, mechanism at play is that the learning of the potential approximation is not based on the sample alone, but also uses information from the target distribution via the way we parametrize our neural network (Section 4.3). This is true for both the original and NSM losses (although the advantage of the NSM loss is to provide a lower variance regression target than the original DSM loss). We note, however, that this second mechanism alone is not sufficient without the help of SMC, again illustrated in Figure 10 and Figure 11.

Cosine scheduler As stated above, we follow the cosine scheduler introduced by Nichol & Dhariwal (2021). We found, as demonstrated in Figure 12 and Figure 13, that the cosine schedule was effective in ensuring the forward SDE converges to the target distribution while regularly spacing the ESS drops across the sampling path.

Number of particles In Figure 14 we show that the normalizing constant estimation error with the naive potential approximation does decrease as the number of particles increases. However, we could not eliminate the error before exceeding the computer memory. We conclude that we must improve our initial potential approximation to obtain feasible results, hence motivating the iterative potential approximation scheme.

Guidance path of NN potential approximation In Figure 15 we demonstrate that using a neural network to correct the naive approximation has the effect of correcting the path of the guidance SDE. Here we use a linear schedule for β_t since the cosine schedule does not allow for analytic roll-out of the SDE.

D.4. Additional results

In Figure 16 we display the normalising constant estimates on all tasks, including the `Brownian` and `Ion` tasks which were omitted from the main text for space.

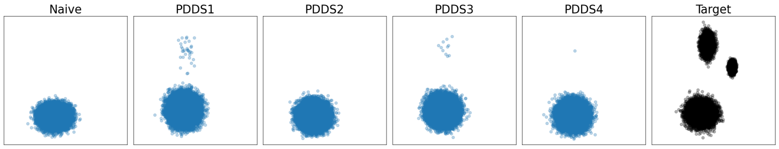

Figure 10: Samples from PDDS without SMC on a 3-mode mixture of Gaussians

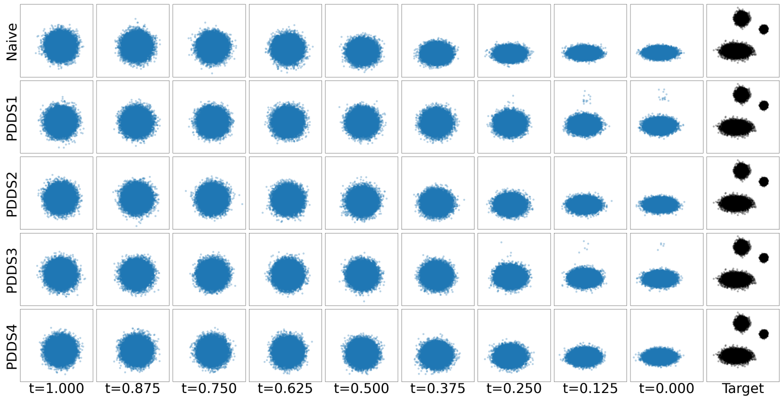

Figure 11: Left to right: marginal distributions of the reverse SDE Equation (7) using the potential approximation at each iteration of PDDS without SMC.

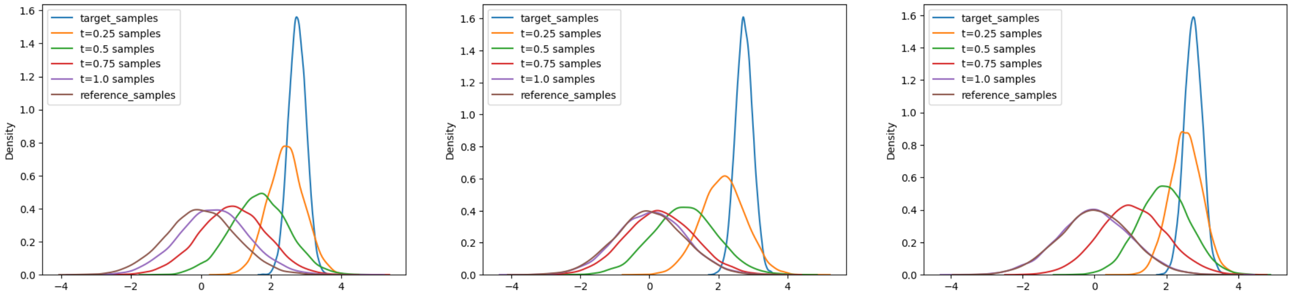

Figure 12: Left: linear schedule, $\beta_f = 8$. Middle: linear schedule: $\beta_f = 12$. Right: cosine schedule.

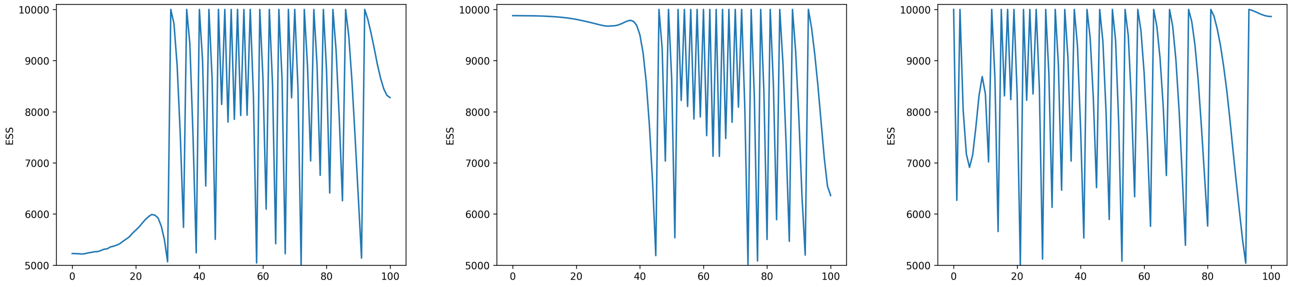

Figure 13: Left: linear schedule, $\beta_f = 8$. Middle: linear schedule: $\beta_f = 12$. Right: cosine schedule.

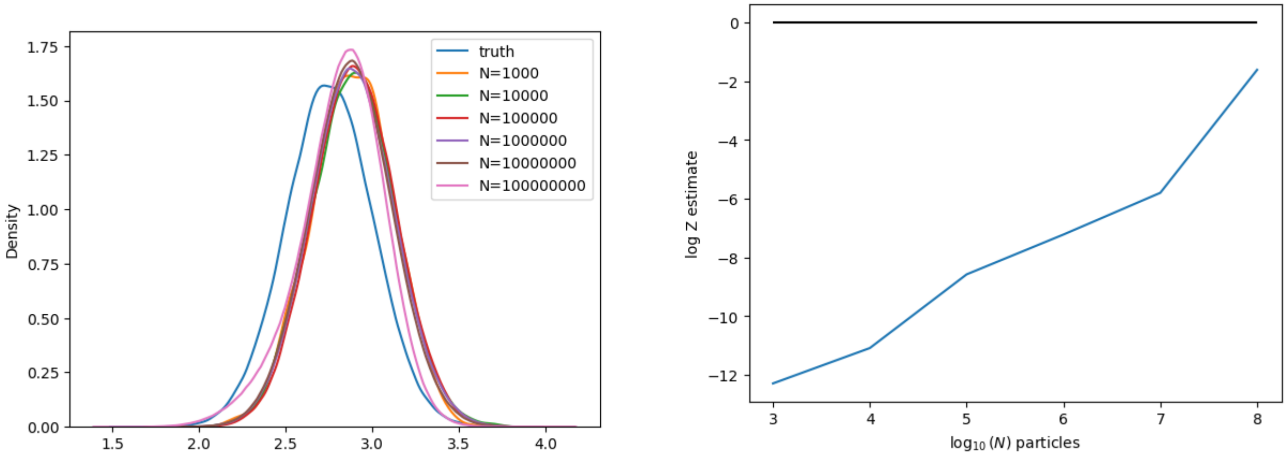

Figure 14: Samples and estimated normalizing constant from PDDS on $\mathcal{N}(2.75, 0.25^2)$ target.

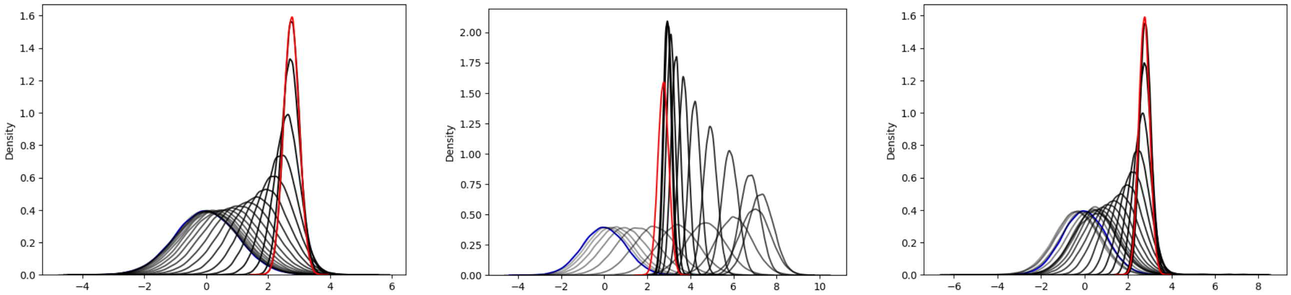

Figure 15: Guidance path using analytic potential function (left), naive potential approximation (middle) and neural network potential approximation (right).

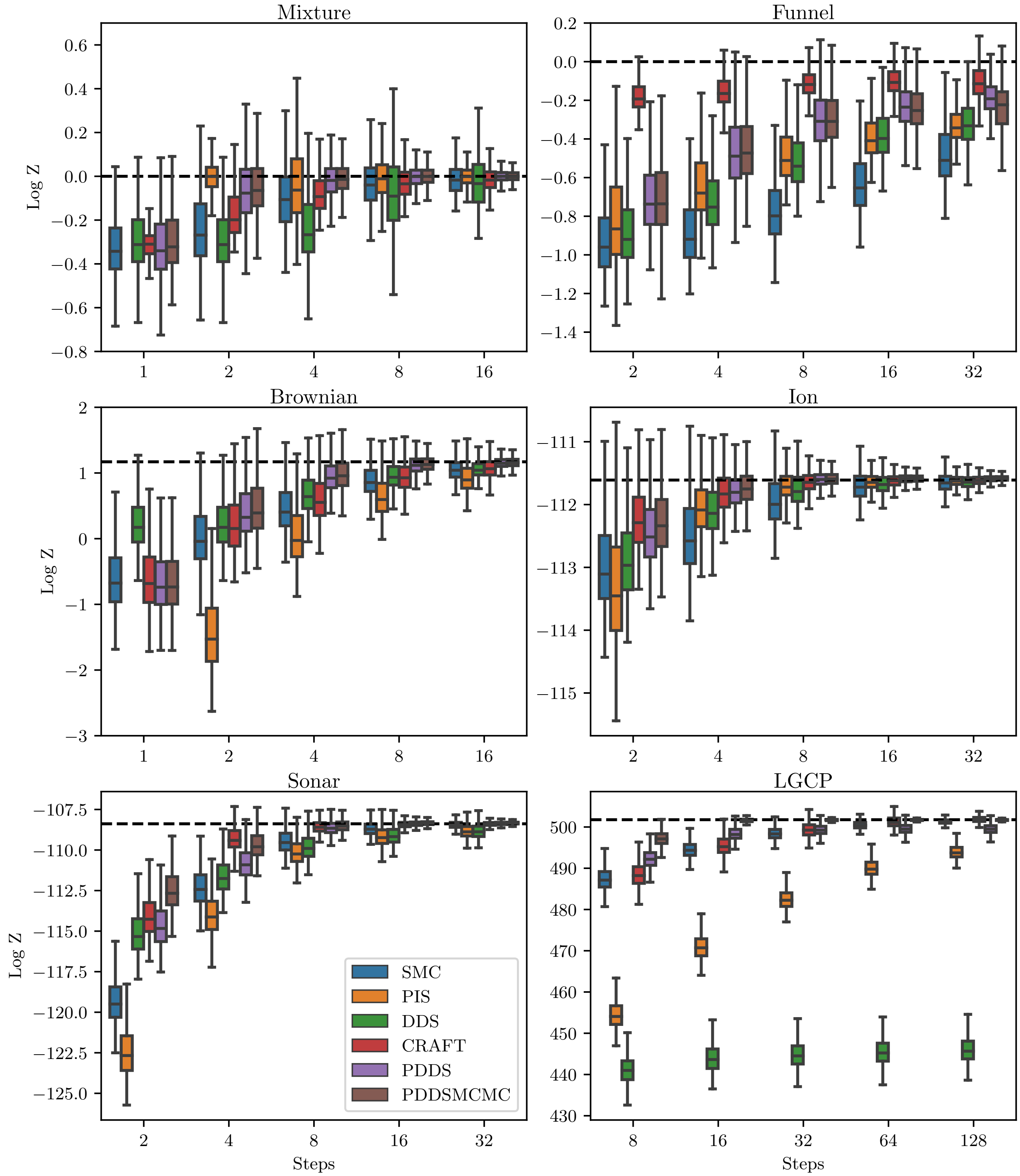

Figure 16: Normalizing constant estimation results on all tasks. Outliers are hidden for clarity. Each box consists of 2000 estimates, coming from 20 training seeds each with 100 evaluation seeds.

In Figure 17 we display the normalising constant estimation results and in Figure 18 the $\mathcal{W}_2^\gamma$ distances for the GMM task in 1, 2, 5, 10, 20 dimensions.

D.5. Uncurated normalizing constant estimates

In Figure 19 we display the normalising constant estimation results of Figure 16 with outliers present. Note that all methods are susceptible to erroneous over-estimation of the normalising constant due to numerical errors an instability. It appears that all methods are equally susceptible to this issue, with no single method displaying more erroneous overestimation than the others. Results on the Gaussian task are displayed separately in Figure 20.

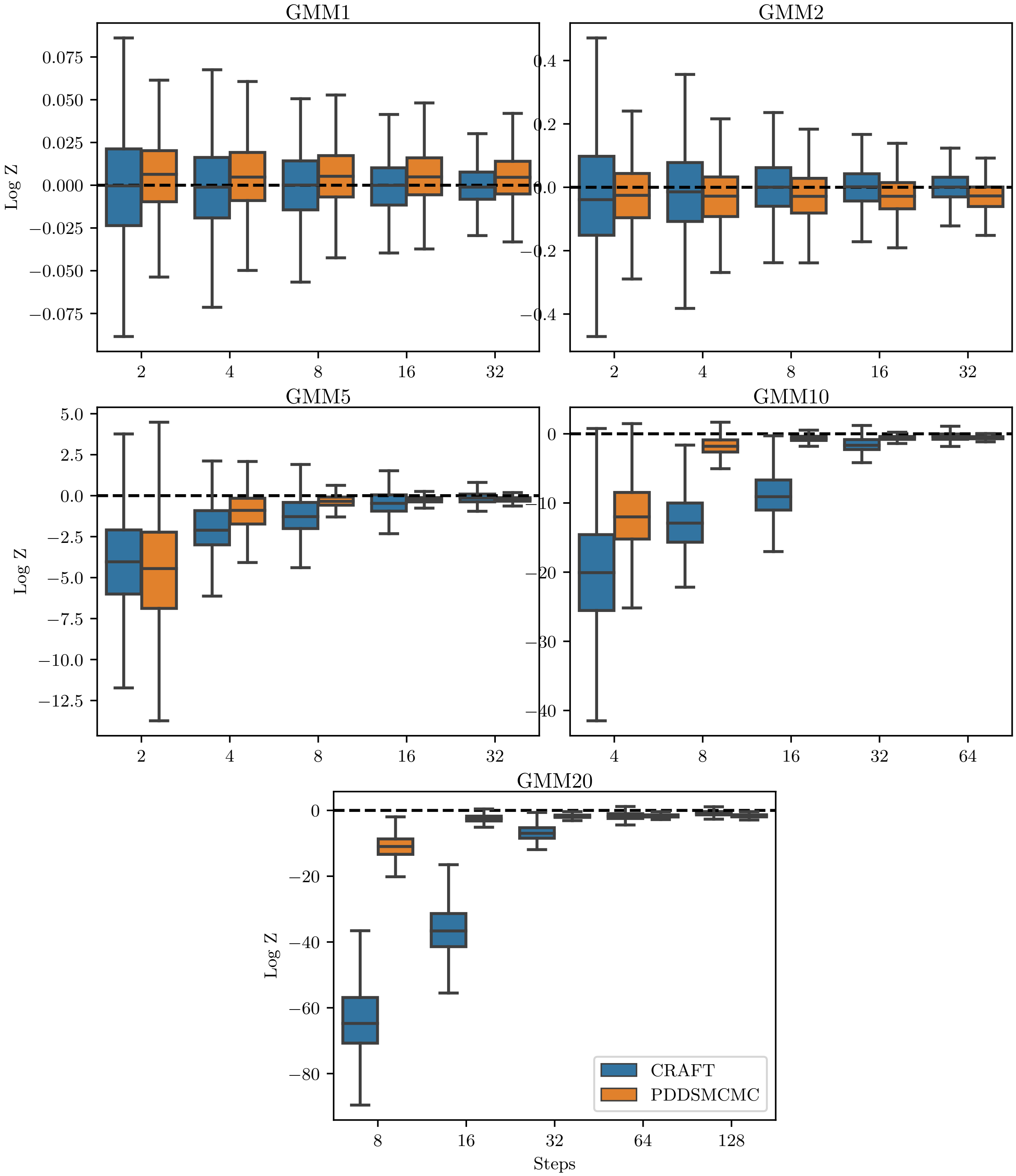

Figure 17: Normalizing constant estimation results on the GMM task in 1, 2, 5, 10 and 20 dimensions. Outliers are hidden for clarity. Each box consists of 1000 estimates coming from 10 training seeds and 100 evaluation seeds.

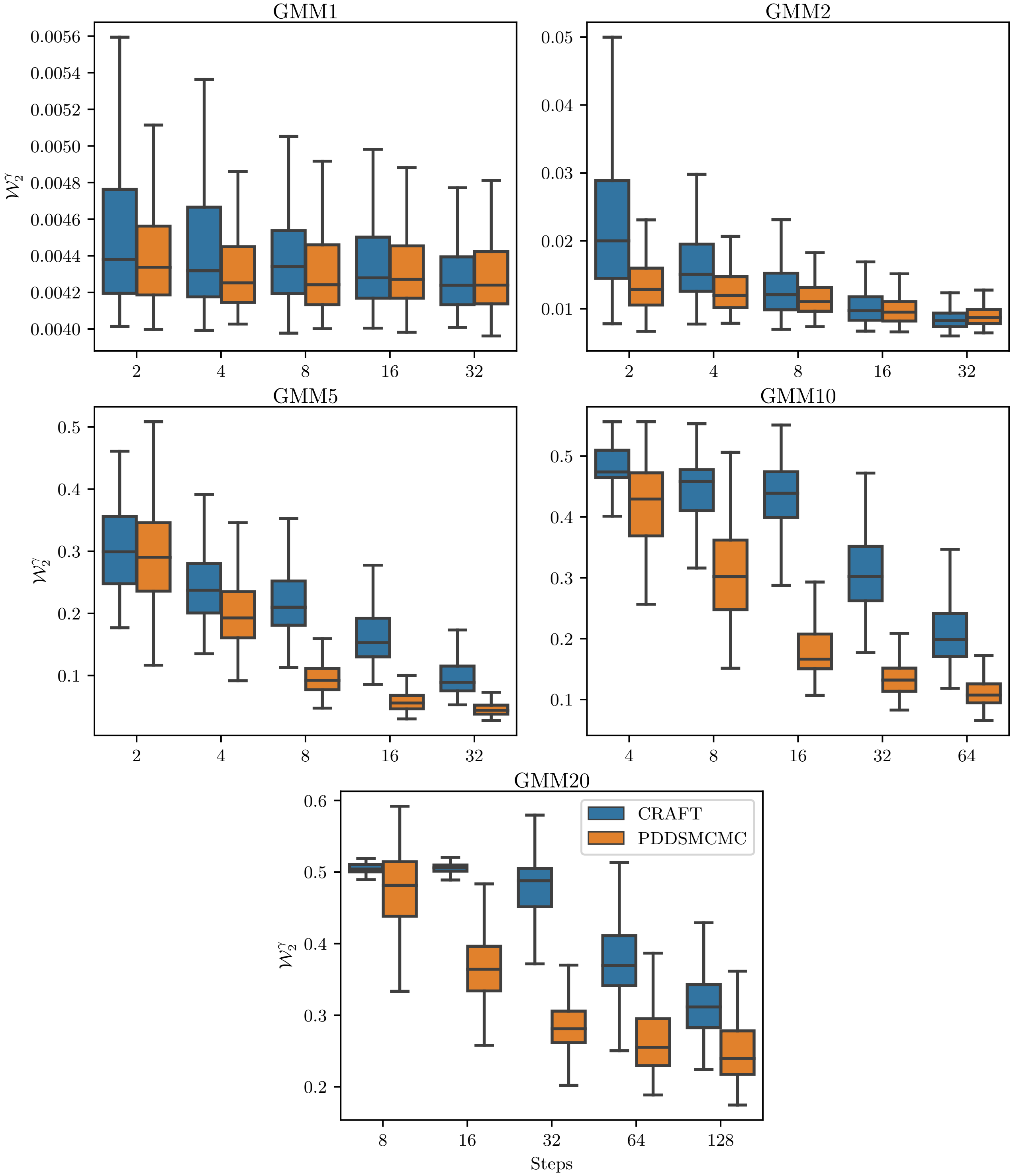

Figure 18: Entropy-regularized Wasserstein-2 distances between samples from the model and target distributions for CRAFT and PDDS-MCMC. Lower is better. Each box consists of 200 values coming from 10 training seeds and 20 evaluation seeds.

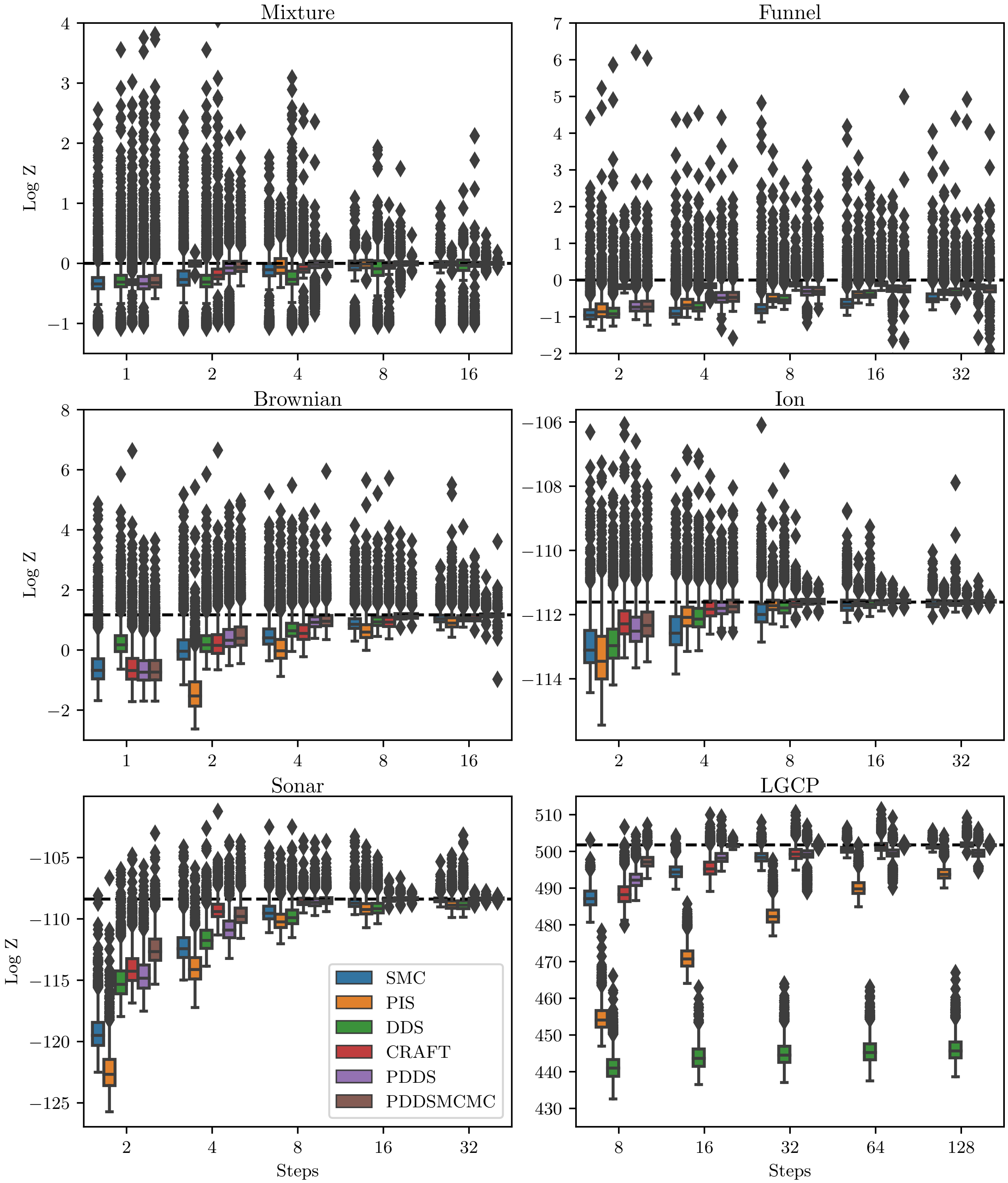

Figure 19: Normalizing constant estimation results on all tasks. Each box consists of 2000 estimates, coming from 20 training seeds each with 100 evaluation seeds.

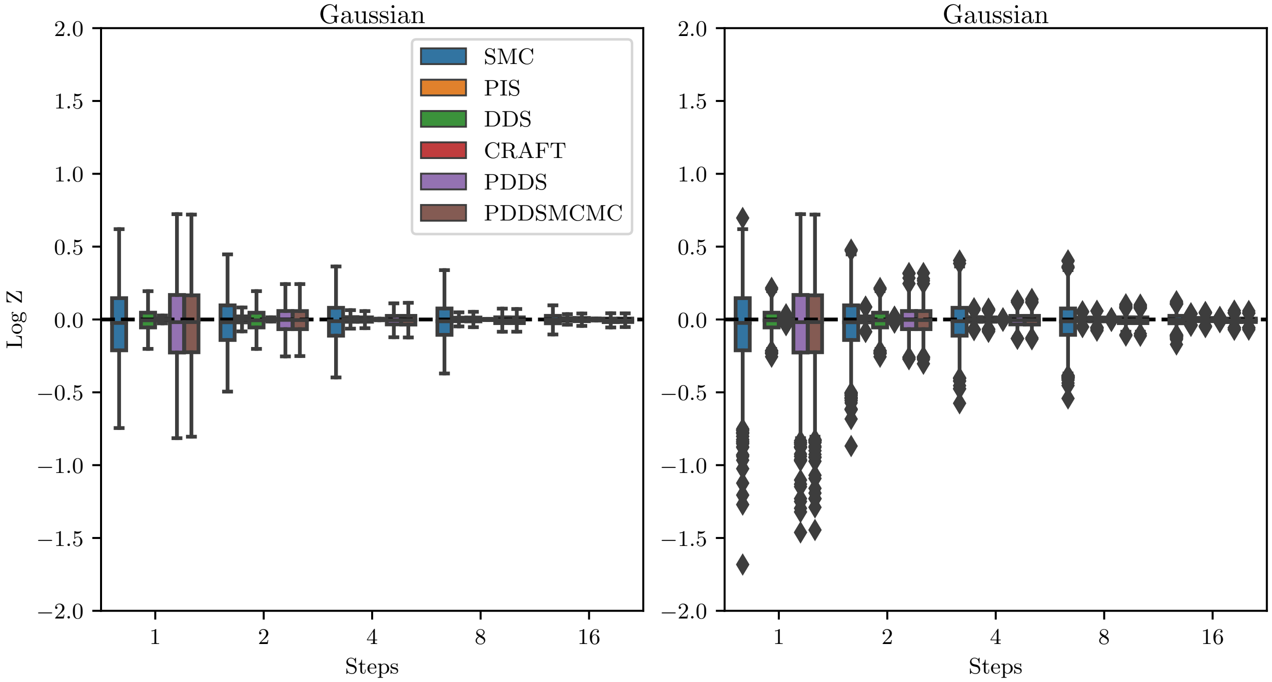

Figure 20: Normalizing constant estimation results on the Gaussian task. Each box consists of 2000 estimates, coming from 20 training seeds each with 100 evaluation seeds. Left: outliers removed, right: uncurated.