A Background: diffusion models and diffusion-based samplers

A.1 Diffusion models

We review common diffusion schedules used in the literature.

Variance-preserving schedule. In Variance-Preserving (VP) schedule [6], the coefficients of the forward SDE in Eq. (1) are given by

$$f(t) = -\frac{1}{2}b(t), \quad g(t) = \sqrt{b(t)}, \tag{16}$$

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

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

for some constants $b_{\min} \approx 0 \ll b_{\max}$. For a large value of $b_{\max}$, the reference distribution is approximately standard normal, $p_1(x) \approx \mathcal{N}(x \mid 0, 1)$.

By Ito's calculus (see e.g. Särkkä and Solin [40]), one can show that the forward transition density $\pi(x_s \mid x_t)$ for $1 \geq s > t \geq 0$ is given by

$$\pi(x_s \mid x_t) = \mathcal{N}\left(x_s \mid \frac{\alpha(s)}{\alpha(t)}x_t, \left(1 - \frac{\alpha^2(s)}{\alpha^2(t)}\right)\mathbb{I}\right),$$

where

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

Variance-exploding schedule. In Variance-Exploding (VE) schedule [7], the forward SDE coefficients in Eq. (1) are given by

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

where $\sigma(t)$ is a non-negative and monotone increasing function with $\sigma(0) = 0$. Commonly used forms of $\sigma(t)$ include polynomial functions, such as linear, quadratic or exponential schedules. When $\sigma(1)$ is large, the reference distribution can be effectively approximated as $\pi_1(x_1) \approx \mathcal{N}(x_1 \mid 0, \sigma^2(1))$.

Similarly to the VP case, one can derive the following parameters of the forward transition kernel with $\alpha(t) = 1$ and $\sigma(t)$ is the given non-negative and monotone function.

A.2 Score identities used for diffusion MC samplers

In addition to the DSI from Eq. (3), one may consider the following identities to form alternative MC estimates of the score, given MC samples of $\pi_t(x_0 \mid x_t)$.

Target score identity (TSI) TSI utilizes known structural properties of the target density, such as estimated gradients, facilitating potentially more accurate and lower-variance score estimations at low-noise settings. Formally, the score function of the noisy target distribution $\pi_t(x)$ under the TSI framework satisfies

$$\nabla_x \log \pi_t(x) = \frac{1}{\alpha(t)} \int \nabla_x \log \pi_t(x_0)\pi_t(x_0 \mid x)dx_0. \tag{17}$$

TSI and its related alternative have been used in several recent studies to improve score estimation accuracy [41, 11, 12, 42].

Mixed score identity (MSI) To balance the advantages of DSI and TSI across different noise regimes, MSI combines both identities through a convex interpolation. Specifically, MSI favors TSI in the low-noise regime and DSI in the high-noise regime, leveraging both strengths. Formally, MSI is a convex combination of TSI and DSI with coefficients $\frac{\alpha(t)^2}{\alpha(t)^2+\sigma(t)^2}$ and $\frac{\sigma(t)^2}{\alpha(t)^2+\sigma(t)^2}$:

$$\nabla_x \log \pi_t(x) = \frac{1}{\alpha(t)^2 + \sigma(t)^2} \int \Big(\alpha(t)\big(x_0 + \nabla_x \log \pi_t(x_0)\big) - x(t)\Big)\pi_t(x_0 \mid x)dx_0. \tag{18}$$

He et al. [43] leverage MSI to train an implicit generative model that produces samples in a single step, demonstrating improved sample quality and efficiency.

Algorithm 2: Importance Sampling (IS)-based score and marginal estimator

Input: Unnormalized prior $\tilde{\pi}(\cdot)$, likelihood parameters $\{\alpha_t, \sigma_t\}$, data x_t, number of importance samples M, and proposal distribution $q(\cdot \mid x_t)$ (default: $\mathcal{N}(\cdot \mid x_t/\alpha_t, \sigma_t^2/\alpha_t^2 \mathbb{I})$)

Output: Score estimate $s(x_t, u_t)$ and marginal estimate $\hat{\pi}(x_t, u_t)$.

1 **for** $m \leftarrow 1$ **to** M **do**

2 Sample $u_t^{(m)} \sim q(\cdot \mid x_t)$

3 Compute importance weights $w^{(m)} \leftarrow \frac{\tilde{\pi}(u_t^{(m)})\mathcal{N}(x_t|\alpha_t u_t^{(m)}, \sigma_t^2 I)}{q(u_t^{(m)})}$

4 Denote auxiliary variable $u_t \leftarrow \{u_t^{(m)}\}_{m=1}^M$

5 Compute score estimate $s(x_t, u_t) \leftarrow \sum_{m=1}^M \frac{w^{(m)}}{\sum_{m'=1}^M w^{(m')}} \cdot \frac{\alpha_t u_t^{(m)} - x_t}{\sigma_t^2}$

6 Compute marginal estimate $\hat{\pi}(x_t, u_t) \leftarrow \frac{1}{M} \sum_{m=1}^M w^{(m)}$.

B Method details

In this section, we discuss several details of RDSMC, including score and marginal estimation strategies in Appendix B.1, the computational complexity in Appendix B.2, and implementation guidelines in Appendix B.3.

B.1 Score and marginal estimation

In this subsection, we present several score and marginal estimators that can be used within RDSMC, which are based on IS, AIS, SMC and rejection sampling. In our main experiments we use AIS-based estimators.

B.1.1 IS-based estimator

The IS-based estimator is summarized in Algorithm 2, which complements the illustration in § 3.1.

Huang et al. [8] consider a similar importance weighted score estimator (see their Eq 5 in Section 3.3) where the proposal $q(\cdot \mid x_t) = \mathcal{N}(\cdot \mid x_t/\alpha_t, \sigma_t^2/\alpha_t^2 \mathbb{I})$ is obtained by reversing the forward noising likelihood. While the IS-based score estimator can be inefficient when t is large, Huang et al. [8] leverages Unadjusted Langevin Algorithm (ULA) with IS estimator for initialization; And when t is close to 0, they are able to quickly obtain rough score estimates via the IS approach.

B.1.2 AIS and SMC-based estimator

Algorithm 3 presents an AIS-based approach [25] to construct the score and marginal estimates. AIS introduces a sequence of intermediate unnormalized distributions, smoothly bridging a tractable initial proposal distribution $q(x_0 \mid x_t)$ to the denoising posterior $\pi(x_0 \mid x_t) \propto \tilde{\pi}(x_0)\mathcal{N}(x_t \mid \alpha_t x_0, \sigma_t^2 \mathbb{I})$. Under a geometric annealing schedule, intermediate targets are $\nu_k(x_0) \propto q(x_0 \mid x_t)^{1-\beta_k} \pi(x_0 \mid x_t)^{\beta_k}$ where $\beta_k = k/n$ for $k = 0, \cdots, n$.

Algorithm 3 begins by drawing a set of particles from the proposal distribution $q(x_0 \mid x_t)$. Then it gradually transitions these particles by running MCMC with a kernel invariant to the intermediate targets $\nu_{k+1}(x_0)$ for $k = 0, \cdots, n-1$. In this work we mainly consider Metropolis-Hasting adjusted Langevin dynamics [MALA, 44] and Hamiltonian Monte Carlo [HMC, 45]. At each intermediate step, the algorithm updates the importance weights of each particle based on the ratio of successive intermediate targets.

The final weighted set of particles provides a consistent estimate of the desired posterior, which is utilized to construct a score estimate $s(x_t, u_t)$, where u_t denotes all the randomness in this procedure. Additionally, these weights provide an unbiased estimate of marginal density $\hat{\pi}(x_t, u_t)$, i.e. $\mathbb{E}_{q(u_t|x_t)}[\hat{\pi}(x_t, u_t)] = Z\pi(x_t)$, where $q(x_t \mid x_t)$ is the sampling distribution of u_t given x_t.

SMC adaption of Algorithm 3. We can adapt Algorithm 3 to an SMC-based procedure by incorporating the resampling mechanism. In this case, the weight update in Line 10 is modified to the

Algorithm 3: Annealed Importance Sampling (AIS)-based Score and Marginal Estimator

Input: Unnormalized prior $\tilde{\pi}(\cdot)$, likelihood parameters $\{\alpha_t, \sigma_t\}$, data x_t, number of importance samples M, number of annealing steps n, initial proposal distribution $q(\cdot \mid x_t)$ (default: $\mathcal{N}(\cdot \mid x_t/\alpha_t, \sigma_t^2/\alpha_t^2\mathbb{I})$), and MCMC-related inputs {number of MCMC steps per transition l, MCMC transition kernel $\mathcal{T}$ and MCMC step size δ_{mcmc}}

Output: Score estimate $s(x_t, u_t)$ and marginal estimate $\hat{\pi}(x_t, u_t)$

1 Set annealing schedule $\beta_k \leftarrow k/n$ for $k = 0, \ldots, n$

2 Set initial target $\nu_1(x_0) \leftarrow q(x_0 \mid x_t)^{1-\beta_1}\tilde{\pi}(x_0)^{\beta_1}\mathcal{N}(x_t \mid \alpha_t x_0, \sigma_t^2\mathbb{I})^{\beta_1}$

3 **for** $m \leftarrow 1$ **to** M **do**

4 Draw initial sample $u_t^{(1,m)} \sim q(\cdot \mid x_t)$

5 Initialize weight $w^{(m)} \leftarrow \frac{\nu_1(u_t^{(1,m)})}{\nu_0(u_t^{(1,m)})}$

6 **for** $k \leftarrow 2$ **to** n **do**

7 Set the current target $\nu_k(x_0) \propto q(x_0 \mid x_t)^{1-\beta_k}\tilde{\pi}(x_0)^{\beta_k}\mathcal{N}(x_t \mid \alpha_t x_0, \sigma_t^2\mathbb{I})^{\beta_k}$

8 **for** $m \leftarrow 1$ **to** M **do**

9 Update sample $u_t^{(k,m)}$ using MCMC transition targeting ν_k starting from $u_t^{(k-1,m)}$

10 Update weight $w^{(m)} \leftarrow w^{(m)} \frac{\nu_k(u_t^{(k,m)})}{\nu_{k-1}(u_t^{(k,m)})}$

11 Denote auxiliary variable $u_t \leftarrow \{\{u_t^{(k,m)}\}_{m=1}^M\}_{k=1}^n$

12 Compute normalized weights $\tilde{w}^{(m)} \leftarrow \frac{w^{(m)}}{\sum_{m'=1}^M w^{(m')}}$

13 Compute score estimate $s(x_t, u_t) \leftarrow \sum_{m=1}^M \tilde{w}^{(m)} \frac{\alpha_t u_t^{(k,m)} - x_t}{\sigma_t^2}$

14 Compute marginal density estimate $\hat{\pi}(x_t, u_t) \leftarrow \frac{1}{M}\sum_{m=1}^M w^{(m)}$

incremental weight computation:

$$w^{(k,m)} \leftarrow \frac{\nu_{k+1}(u_t^{(m)})}{\nu_k(u_t^{(m)})}$$

and the initial weight in Line 5 is denoted by $w^{(1,m)}$, for $m = 1, \cdots, M$. With this adjustment, we resample particles $\{u_t^{(m)}\}_{m=1}^M$ in the beginning of each iteration k based on the previous weights $\{w^{(k-1,m)}\}_{m=1}^M$ for $k = 2, \cdots, n$. The marginal estimate in Line 14 becomes

$$\hat{\pi}(x_t, u_t) \leftarrow \frac{1}{n}\sum_{k=1}^n \frac{1}{M}\sum_{m=1}^M w^{(k,m)}.$$

The other aspect of Algorithm 3 remains the same.

B.1.3 Rejection sampling (RS)-based estimator

In Algorithm 4, we present a rejection sampling (RS)-based approach, building on prior work of He et al. [10] which propose an RS-based score estimator and develop theoretical guarantees for the resulting diffusion MC sampler. We extend their approach to additionally provide an unbiased marginal density estimate alongside the original score estimate. In principle, this procedure can be used within RDSMC (Algorithm 1) as a replacement for the AIS scheme (Algorithm 3). Future work may explore this direction particularly in low-dimensional settings, as He et al. [10] demonstrate advantages over other score estimators in a non-SMC context.

Notably, rejection sampling requires access to a constant C that upper bounds the ratio of the unnormalized target to the proposal. He et al. [10] use Newton's method to find a proxy of C.

B.1.4 Proposal choices in IS/AIS/SMC/RS-based score estimators.

Algorithms 2 to 4 require specifying an (initial) proposal distribution $q(x_0 \mid x_t)$. The choice of this proposal is flexible and should ideally approximate the true posterior $\pi(x_0 \mid x_t)$; the closer it is, the more effective the resulting estimator.

Algorithm 4: Rejection Sampling (RS)–based score and marginal estimator

Input: Unnormalized prior $\tilde{\pi}(\cdot)$, likelihood parameters $\{\alpha_t, \sigma_t\}$, data x_t, number of accepted samples M, proposal $q(\cdot \mid x_t)$ (default: $\mathcal{N}\left(\cdot \mid x_t/\alpha_t, \alpha_t^2/\sigma_t^2 \mathbb{I}\right)$), and an upper bound $C \geq \sup_{u_t} \tilde{\pi}(u_t)\mathcal{N}(x_t \mid \alpha_t u_t, \sigma_t^2 I)/q(u_t)$

Output: Score estimate $s(x_t, u_t)$ and marginal estimate $\hat{\pi}(x_t, u_t)$.

$m, i \leftarrow 1, 1$
while $m \leq M$ **do**
  Sample proposal $v_i \sim q(\cdot \mid x_t)$
  Compute acceptance ratio $r \leftarrow \frac{\tilde{\pi}(v_i)\mathcal{N}(x_t \mid \alpha_t v_i, \sigma_t^2 I)}{C\, q(v_i)}$
  Draw $v_i \sim \mathrm{Uniform}(0, 1)$
  if $v < r$ **then**
    Accept sample $u_t^{(m)} \leftarrow v_i$
    $m \leftarrow m + 1$
  $i \leftarrow i + 1$
$M_{\text{total}} \leftarrow i - 1$
Denote auxiliary variable $u_t \leftarrow \{v_i, v_i\}_{i=1}^{M_{\text{total}}}$
Compute score estimate: $s(x_t, u_t) \leftarrow \frac{1}{M} \sum_{m=1}^{M} \frac{\alpha_t u_t^{(m)} - x_t}{\sigma_t^2}$
Compute marginal estimate: $\hat{\pi}(x_t, u_t) \leftarrow C \frac{M}{M_{\text{total}}}$

In Algorithms 2 to 4, the default initial proposal is obtained by reversing the forward noising likelihood, suggested by prior work [8, 12, 43]. This proposal is effective when the likelihood signal is strong—i.e., when t is close to 0—but becomes less efficient as t increases and the signal weakens.

Alternative strategies include using a Gaussian proposal centered at the current x_t. If additional information about the target distribution is available—such as its mean or covariance—it can be incorporated into the proposal. For instance, one may use a Laplace approximation of the target to construct an approximate posterior, which can be used as a sensible proposal.

B.1.5 Score estimator based on other score identities

In Eq. (5) of the main paper, we derive the score estimator based on the DSI in Eq. (3). Alternative score estimators can be derived based on other score identities.

Based on TSI in Eq. (17), we obtain another estimate:

$$\nabla_{x_t} \log p(x_t) \approx s(x_t, u_t) := \sum_{m=1}^{M} \frac{w^{(m)}}{\sum_{m'=1}^{M} w^{(m')}} \cdot \frac{\nabla_{u_t} \log q(u_t^{(m)})}{\alpha_t}, \tag{19}$$

where $\{w^{(m)}, u_t{}^{(m)}\}_{m=1}^{M}$ is a weighted system approximating the posterior $\pi(x_0 \mid x_t)$ using an IS/AIS/SMC procedure.

Or, given MSI in Eq. (18), we have:

$$\nabla_{x_t} \log p(x_t) \approx s(x_t, u_t) := \sum_{m=1}^{M} \frac{w^{(m)}}{\sum_{m'=1}^{M} w^{(m')}} \cdot \Big(\alpha_t\big(u_t^{(m)} + \nabla_{u_t} \log q(u_t^{(m)})\big) - x_t\Big). \tag{20}$$

We primarily consider DSI and TSI-based estimators in our experiments.

B.2 Computational complexity

The total computational cost of RDSMC scales linearly with the number of diffusion discretization steps T, the number of particles N, and the cost of each score estimation step.

Specifically, for AIS-based score estimator as described in Algorithm 3, the score estimation cost scales linearly with the number of importance samples M, the number of AIS transition steps n, and

the number of MCMC steps per transition l. The vanilla IS-based score estimator in Algorithm 2 is a special case with $n = 1$ and $l = 1$. Therefore the cost of score estimation per SMC particle per time step $\mathcal{O}(Mnl)$. Consequently, the overall complexity for RDSMC is $\mathcal{O}(NT \cdot Mnl)$.

Among these factors, the operations involving discretization steps T, AIS transition steps n, and MCMC steps l must be processed sequentially due to their time-dependent structure. In contrast, computations over the particles N and the importance samples M are embarrassingly parallel and can be efficiently distributed across parallel compute resources.

B.3 Implementation guidelines

We discuss practical implementation guidelines for RDSMC, including the resampling strategy in Appendix B.3.1, score clipping in Appendix B.3.2 and hyperparameter choices in Appendix B.3.3.

B.3.1 SMC resampling strategy

While the resampling step in RDSMC is crucial for steering particles toward high target probability region, it introduces extra variance in the short term. We discuss several strategies to reduce this added variance.

First, we use systematic resampling, which introduces correlations among resampling indices [19, Chapter 2.2.1].

We also consider start resampling at a later stage of the sampling trajectory. At early steps, the Gaussian likelihood provides weaker conditional information, making posterior inference more challenging [8, 9]. Consequently, both the score estimates and marginal density estimates can be significantly biased or exhibit high variance, especially under limited compute budgets. As a result, the intermediate target distributions in early steps may not be sufficiently informative to enable effective resampling. To mitigate this issue, we delay resampling until a designated step $t_{\text{start-resampling}} < T$, which we treat as a tunable hyperparameter.

For timestep $t < t_{\text{start-resampling}}$, we adopt an effective sample size (ESS)-based criterion to decide whether resampling should be performed at each step – a common strategy in SMC literature [19, Chapter 2.2.2]. The ESS at step t is computed as

$$ESS_t := \frac{\left(\sum_{i=1}^N w_t^{(i)}\right)^2}{\sum_{i=1}^N \left(w_t^{(i)}\right)^2}. \tag{21}$$

We trigger resampling only when the normalized ESS, computed as ESS_t/N,is below a threshold κ_{ess}, another hyperparameter which we fix at 0.3 throughout experiments.

When resampling is skipped (i.e. for $t > t_{\text{start-resampling}}$ or when the normalized ESS is above κ_{ess}), we must adjust the computation of the intermediate weights to ensure the unbiasedness of the log normalization constant estimate [19, Chapter 2.2.3]. For iterations where resampling is not performed, the weight update in Line 13 of Algorithm 1 is replaced by

$$w_t^{(i)} \leftarrow \frac{\bar{w}_{t+1}^{(i)}}{1/N} \cdot \frac{\hat{\pi}_t^{(i)} p(x_{t+1}^{(i)} \mid x_t^{(i)})}{\hat{\pi}_{t+1}^{(i)} q(x_t^{(i)} \mid x_{t+1}^{(i)}, u_{t+1}{}^{(i)})} \tag{22}$$

where $\{\bar{w}_{t+1}^{(i)}\}_{i=1}^N$ are the normalized weights in the previous iteration.

B.3.2 Score clipping

To improve numerical stability, one may consider clipping the score estimates within a pre-specified range. Note that score clipping does not affect our theoretical guarantees provided that regularity conditions hold, as the score estimate is only part of the proposal mechanism.

As observed in prior work [46], score estimates produced by the TSI estimator defined in Eq. (17) can exhibit high variance or have large magnitudes, especially during early steps. Akhound-Sadegh et al. [12] apply score norm clipping when training diffusion samplers where the loss function is based on the TSI. Inspired by their work, we also clip the norm of the score estimates to a maximum of 20

when we use the TSI-based score estimates. In contrast, we do not apply clipping when using the DSI-based score estimator.

B.3.3 Hyperparameter choices

The main hyperparameters of RDSMC are:

Diffusion-related hyperparameters.

- Diffusion parameterization: we use the VP diffusion schedule described in Eq. (16) in this work.
- Time discretization strategy: we use a uniform time discretization grid with $T = 100$ steps, though more informative strategies can be incorporated—such as the SNR-adapted scheme proposed by Grenioux et al. [9].

MC score estimation-related hyperparameters.

- Score estimator type: we use AIS-based score estimator (Algorithm 3), with vanilla IS (Algorithm 2) as a special case. We mainly leverage the DSI (Eq. (5)) to estimate the score in most experiments, while we also use the TSI (Eq. (19)) in Bayesian logistic regression tasks.
- MCMC transition kernel: MALA or HMC.
- M: Number of importance samples used in Monte Carlo score estimation
- n: Number of intermediate distributions, i.e. number of AIS steps; for IS $n = 1$
- δ_{mcmc}: Step size used in the MCMC transition kernel (not used in IS). We manually tune this parameter in the main experiments while we use adaptive step size in ablation studies; see discussion below.
- l : Number of MCMC steps per AIS transition (not used in IS). We use $l = 1$ in all experiments.

SMC-related hyperparameters.

- $t_{\text{start-resampling}}$: Starting time for resampling within the SMC trajectory. No resampling is performed for $t/T > t_{\text{start-resampling}}$.
- κ_{ess}: Normalized effective sample size threshold triggering resampling. When $\kappa_{\text{ess}} = 0$, no resampling is applied throughout, which corresponds to RDSMC (IS) if a final-step weighting is applied and RDSMC (Proposal) otherwise. We set $\kappa_{\text{ess}} = 0.3$ in all experiments for RDSMC.

In the main experiments, we choose score estimator and MCMC kernel types by a coarse grid search, and tune $M, n, \delta_{\text{mcmc}}$ and $t_{\text{start-resampling}}$ in a range of values, using validation metrics. The hyperparameter values are summarized Table 3; see details in Appendix E.1.2.

In the ablation experiments in Appendix E.5, we fix most hyperparameters to reduce the degrees of freedom. We use $T = 100, M = 100, n = 50, m = 1, \kappa_{\text{ess}} = 0.3$, and AIS-based score estimator leveraging the DSI, LG as the MCMC kernel. We also use an *adaptive* MCMC step size δ_{mcmc}, as in Grenioux et al. [9]. When the acceptance rate is too high ($> 76\%$) we increase the step size by 3%, when it is $< 74\%$, we decrease step size by 3%, and otherwise we keep the current step size. Only an initial step size value needs to be specified and then it is adpatively adjusted during the sampling process. The *only* hyperparameter in RDSMC that requires tuning is $t_{\text{start-resampling}}$, which we vary in $\{0, 0.1, \cdots, 0.9, 1.0\}$. We observe consistent performance and therefore in practice we would recommend users to start with something similar to this configuration.

C Theoretical guarantees of RDSMC

We formally state the theorem providing sufficient conditions for asymptotic accuracy of RDSMC and unbiasedness of the estimate of the normalization constant. Our proof strategy for asymptotic

accuracy is adapted from Wu et al. [30] and the proof for unbiasedness follows the techniques presented in Naesseth et al. [19].

Theorem 2. *Assume*

(a) the first marginal estimate $\hat{\pi}(x_T, u_T)$, and the ratios of subsequent ratio $\hat{\pi}(x_t, u_t)/\hat{\pi}(x_{t+1}, u_{t+1})$ are positive and bounded,

(b) the score estimate $s(x_t, u_t)$ is bounded, and x_t has compact support,

(c) variance increases in the forward noising kernel, i.e. for each t, $g_{t+1}^2 > g_t^2$.

Then the RDSMC algorithm described in Algorithm 1 provides a consistent estimator of the target distribution $\pi(x_0)$ as particle size $N \to \infty$ and an unbiased estimator of the normalization constant Z for any $N \geq 1$. More specifically,

(a) Let $\mathbb{P}_N = \sum_{i=1}^N w_0^{(i)} \delta_{x_0^{(i)}}$ for weighted particles $\{(x_0^{(i)}, w_0^{(i)})\}_{i=1}^N$ returned by Algorithm 1 with N particles. $\mathbb{P}_N$ converges setwise to $\pi(x_0)$ with probability one, that is for every set A, $\lim_{N\to\infty} \mathbb{P}_N(A) = \int_A \pi(x_0) dx_0$.

(b) For the estimate for the normalization constant $\hat{Z}$ returned by Algorithm 1, we have $\mathbb{E}\hat{Z} = Z$.

The assumptions of Theorem 2 are typically satisfied in standard implementations of SMC samplers:

- Assumption (a) can be satisfied by the clipping of the marginal estimates $\hat{\pi}(x_t, u_t)$ within the range $[c, C]$ for some constants $c, C > 0$.
- Assumption (b) by clipping scores estimates as well. Additionally, the compactness of the state space x_t is a standard and commonly adopted assumption in the SMC literature.
- Assumption (c) is mild and typically satisfied in practice. In both VE and VP noising schedules commonly used in diffusion models, the variance of the forward kernel increases with time.

Proof of Theorem 2: We first prove the asymptotic exactness of our sampler and then prove the unbiasedness of the normalization constant estimate.

First, Theorem 3 characterizes a set of conditions under which SMC algorithms converge. We restate this result below in our own notation.

Theorem 3 (Chopin and Papaspiliopoulos [20] – Proposition 11.4)**.** *Let $\{(x_0^{(i)}, w_0^{(i)})\}_{i=1}^N$ be the particles and weights returned at the last iteration of a sequential Monte Carlo algorithm with N particles using multinomial resampling. If each weighting function w_t is positive and bounded, then for every bounded, ν_0- measurable function ϕ of x_t*

$$\lim_{N\to\infty} \sum_{i=1}^N w_0^{(i)} \phi(x_0^{(i)}) = \int \phi(x_0)\nu_0(x_0) dx_0.$$

with probability one.

An immediate consequence of Theorem 3 is the setwise convergence of the discrete measures. To apply Theorem 3to show asymptotic accuracy, it is sufficient to show that the weights at each step are upper bounded, as they are defined through probabilities and hence are positive. Since there is a finite number of steps T, it suffices to show that each w_t is bounded.

The initial weight is defined by

$$w_T = \frac{\hat{\pi}(x_T, u_t)}{q(x_T)},$$

which is bounded by Assumption (a).

The intermediate weights are defined by Eq. (14) such that

$$w_t = \frac{\hat{\pi}(x_t, u_t)\, \pi(x_{t+1} \mid x_t)}{\hat{\pi}(x_{t+1}, u_{t+1})\, q(x_t \mid x_{t+1}, u_{t+1})}.$$

We first decompose the weights by

$$\log w_t = \log \frac{\hat{\pi}(x_t, u_t)}{\hat{\pi}(x_{t+1}, u_{t+1})} + \log \frac{\pi(x_{t+1} \mid x_t)}{q(x_t \mid x_{t+1}, u_{t+1})}.$$

The boundedness of $\log \frac{\hat{\pi}(x_t, u_t)}{\hat{\pi}(x_{t+1}, u_{t+1})}$ is guaranteed by Assumption (a). We now show the boundedness of $\log \frac{\pi(x_{t+1}|x_t)}{q(x_t|x_{t+1}, u_{t+1})}$. First, note that

$$\pi(x_{t+1} \mid x_t) = \mathcal{N}(x_{t+1} \mid x_t + f_t x_t \delta, g_t^2 \delta),$$

and

$$q(x_t \mid x_{t+1}, u_{t+1}) = \mathcal{N}(x_t \mid x_{t+1} - \left[f_{t+1} x_{t+1} - g_{t+1}^2 s(x_{t+1}, u_{t+1})\right] \delta, g_{t+1}^2 \delta).$$

Let $\hat{\mu}_p = x_{t+1} - f_t x_t \delta$ and $\hat{\mu}_q = x_{t+1} - \left[f_{t+1} x_{t+1} - g_{t+1}^2 s(x_{t+1}, u_{t+1})\right] \delta$. We can show that $\|\hat{\mu}_p - \hat{\mu}_q\| = \|f_{t+1} x_{t+1} - f_t x_t - g_{t+1}^2 s(x_{t+1}, u_{t+1})\| \delta$ is bounded , which follows from Assumption (b). The log-ratio then simplifies as

$$\log \frac{\pi(x_{t+1} \mid x_t)}{q(x_t \mid x_{t+1}, u_{t+1})} \tag{23}$$

$$= \log \frac{|2\pi g_t^2 I|^{-1/2} \exp\{-(2g_t^2)^{-1} \|x_t - \hat{\mu}_p\|^2\}}{|2\pi g_{t+1}^2 I|^{-1/2} \exp\{-(2g_{t+1}^2)^{-1} \|x_t - \hat{\mu}_q\|^2\}} \tag{24}$$

Rearrange and let $C = \log |2\pi g_t^2 I|^{-1/2} / |2\pi g_{t+1}^2 I|^{-1/2}$ (25)

$$= -\frac{1}{2} \left[g_t^{-2} \|x_t - \hat{\mu}_p\|^2 - g_{t+1}^{-2} \|x_t - \hat{\mu}_q\|^2\right] + C \tag{26}$$

Expand and rearrange $\|x_t - \hat{\mu}_q\|^2 = \|x_t - \hat{\mu}_p\|^2 + 2\langle \hat{\mu}_p - \hat{\mu}_q, x_t - \hat{\mu}_p \rangle + \|\hat{\mu}_p - \hat{\mu}_q\|^2$ (27)

$$= -\frac{1}{2} \left[(g_t^{-2} - g_{t+1}^{-2}) \|x_t - \hat{\mu}_p\|^2 - 2 g_{t+1}^{-2} \langle \hat{\mu}_p - \hat{\mu}_q, x_t - \hat{\mu}_p \rangle - g_{t+1}^{-2} \|\hat{\mu}_p - \hat{\mu}_q\|^2\right] + C \tag{28}$$

Let $C' = C + \frac{1}{2} g_{t+1}^2 \|\hat{\mu}_p - \hat{\mu}_q\|^2$ and rearrange. Note that $\|\hat{\mu}_p - \hat{\mu}_q\|^2 < \infty$ (29)

$$= -\frac{1}{2} (g_t^{-2} - g_{t+1}^{-2}) \|x_t - \hat{\mu}_p\|^2 + g_{t+1}^{-2} \langle \hat{\mu}_p - \hat{\mu}_q, x_t - \hat{\mu}_p \rangle + C' \tag{30}$$

By Cauchy-Schwarz, (31)

$$\leq -\frac{1}{2} (g_t^{-2} - g_{t+1}^{-2}) \|x_t - \hat{\mu}_p\|^2 + g_{t+1}^{-2} \|\hat{\mu}_p - \hat{\mu}_q\| \cdot \|x_t - \hat{\mu}_p\| + C' \tag{32}$$

By Assumption (c), $g_t^{-2} - g_{t+1}^{-2} > 0$, (33)

$$\leq \frac{1}{2} \frac{(g_{t+1}^{-2} \|\hat{\mu}_p - \hat{\mu}_q\|)^2}{g_t^{-2} - g_{t+1}^{-2}} + C' \tag{34}$$

$$= \frac{g_{t+1}^{-4}}{2(g_t^{-2} - g_{t+1}^{-2})} \|\hat{\mu}_p - \hat{\mu}_q\|^2 + C' \tag{35}$$

$$\leq C''. \tag{36}$$

The final line follows from the boundedness of $\|\hat{\mu}_p - \hat{\mu}_q\|$. The above derivation shows that each w_t is bounded, which concludes the proof for the asymptotic exactness of our sampler.

Then we show the unbiasedness of the normalization constant estimate $\hat{Z}$. Let a_t^i be the index of $x_t^{(i)}$ and $u_t^{(i)}$ before resampling. With that, $\hat{Z}$ is defined by

$$\hat{Z} = \prod_{t=0}^{T} \frac{1}{N} \sum_{i=1}^{N} w_t^{(i)}, \quad \text{where } w_t^{(i)} = \frac{\hat{\pi}_t^{(i)} \pi(x_{t+1}^{(i)} \mid x_t^{a_t^i})}{\hat{\pi}_{t+1}^{(i)} q(x_t^{a_t^i} \mid x_{t+1}^{(i)}, u_{t+1}^{(i)})}, \; w_T^{(i)} = \frac{\hat{\pi}_T^{(i)}}{\hat{\pi}(x_T^{a_t^i})}.$$

Denote the normalized weight by $\widetilde{w_t}^{(i)} = \frac{w_t^{(i)}}{\sum_{i=1}^N w_t^{(i)}}$. The distribution of all random variables generated by the SMC method is then given by

$$
\begin{aligned}
&Q(x_{0:T}, u_{0:T}, a_{0:T}) \\
&= \pi(x_t, u_T) p(x_{0:T-1}, u_{0:T-1}, a_{0:T}) \\
&= \pi(x_t^{a_T}, {u_T}^{a_T}) \pi(x_t, u_T \mid x_T^{a_T}, {u_T}^{a_T}) \prod_{t=0}^{T-1} p(x_t, u_t \mid x_t^{a_t}, {u_t}^{a_t}) q(x_t^{a_t} \mid x_{t+1}, u_{t+1}) q({u_t}^{a_t} \mid x_t^{a_t}), \\
&= \pi(x_t^{a_T}) \Big(\prod_{i=1}^N \frac{w_T^{(i)}}{\sum_{i=1}^N w_T^{(i)}} \Big) \prod_{i=1}^N \Big[\Big(\prod_{t=0}^{T-1} \frac{w_t^{(i)}}{\sum_{i=1}^N w_t^{(i)}} q(x_t^{a_t^i} \mid x_{t+1}^{(i)}, u_{t+1}^{(i)}) \Big) \Big] \\
&\Big[q({u_T}^{a_T} \mid x_T^{a_T}) \prod_{t=0}^{T-1} q({u_t}^{a_t} \mid x_t^{a_t}) \Big].
\end{aligned}
$$

By $w_t^{(i)} = \frac{\hat{\pi}_t^{(i)} \pi(x_{t+1}^{(i)} | x_t^{a_t^i})}{\hat{\pi}_{t+1}^{(i)} q(x_t^{a_t^i} | x_{t+1}^{(i)}, u_{t+1}^{(i)})}$ and $w_T^{(i)} = \frac{\hat{\pi}_T^{(i)}}{q(x_T^{a_T^i})}$, we have:

$$
\begin{aligned}
&Q(x_{0:T}, u_{0:T}, a_{0:T}) \\
&= \Big(\prod_{t=0}^T \sum_{i=1}^N w_t^{(i)} \Big)^{-1} \hat{\pi}_0^{(1)} \Big(\prod_{t=0}^{T-1} \pi(x_{t+1}^{(1)} | x_t^{a_t^1}) \Big) \\
&\prod_{i=2}^N \hat{\pi}_T^{(i)} \prod_{i=2}^N \Big[\Big(\prod_{t=0}^{T-1} \widetilde{w_t}^{(i)} q(x_t^{a_t^i} \mid x_{t+1}^{(i)}, u_{t+1}^{(i)}) \Big) \Big] \Big[q({u_T}^{a_T} \mid x_T^{a_T}) \prod_{t=0}^{T-1} q({u_t}^{a_t} \mid x_t^{a_t}) \Big],
\end{aligned}
$$

where we plug in $w_t^{(1)}$ and rearrange terms.

By multiplying $\hat{Z}$, we have:

$$
\begin{aligned}
\hat{Z} Q(x_{0:T}, u_{0:T}, a_{0:T}) &= \hat{Z} \Big(\prod_{t=0}^T \sum_{i=1}^N w_t^{(i)} \Big)^{-1} \hat{\pi}_0^{(1)} \Big(\prod_{t=0}^{T-1} \pi(x_{t+1}^{(1)} | x_t^{a_t^1}) \Big) \\
&\prod_{i=2}^N \Big[\hat{\pi}_T^{(i)} \Big(\prod_{t=0}^{T-1} \widetilde{w_t}^{(i)} q(x_t^{a_t^i} \mid x_{t+1}^{(i)}, u_{t+1}^{(i)}) \Big) \Big] \Big[q({u_T}^{a_T} \mid x_T^{a_T}) \prod_{t=0}^{T-1} q({u_t}^{a_t} \mid x_t^{a_t}) \Big] \\
&= N^{-T} \hat{\pi}_0^{(1)} q(u_0^{a_0^1} \mid x_0^{a_0^1}) \Big(\prod_{t=0}^{T-1} \pi(x_{t+1}^{(1)} | x_t^{a_t^1}) q({u_{t+1}}^{a_{t+1}^1} \mid x_{t+1}^{a_{t+1}^1}) \Big) \\
&\prod_{i=2}^N \Big[\hat{\pi}_T^{(i)} q(u_t^{a_T^i} \mid x_t^{a_T^i}) \Big(\prod_{t=0}^{T-1} \widetilde{w_t}^{(i)} q(x_t^{a_t^i} \mid x_{t+1}^{(i)}, u_{t+1}^{(i)}) q({u_{t+1}}^{a_{t+1}^{(i)}} \mid x_{t+1}^{a_{t+1}^{(i)}}) \Big) \Big],
\end{aligned}
$$

where the first equality follows from plugging in Q and the second follows from plugging in $\hat{Z}$.

The expectation of the normalization constant estimate can be written as

$$
\begin{aligned}
\mathbb{E}_Q(\hat{Z}) =& N^{-T} \sum_{a_{0:T}} \int \int \hat{\pi}_0^{(1)} q(u_0^{a_0^1} \mid x_0^{a_0^1}) \Big(\prod_{t=0}^{T-1} \pi(x_{t+1}^{(1)} | x_t^{a_t^1}) q({u_{t+1}}^{a_{t+1}^1} \mid x_{t+1}^{a_{t+1}^1}) \Big) \\
&\prod_{i=2}^N \Big[\hat{\pi}_T^{(i)} q(u_t^{a_T^i} \mid x_t^{a_T^i}) \Big(\prod_{t=0}^{T-1} \widetilde{w_t}^{(i)} q(x_t^{a_t^i} \mid x_{t+1}^{(i)}, u_{t+1}^{(i)}) q({u_{t+1}}^{a_{t+1}^{(i)}} \mid x_{t+1}^{a_{t+1}^{(i)}}) \Big) \Big] dx_{0:T} du_{0:T} \\
=& N^{-T} \sum_{a_{0:T}^1} \int \int \hat{\pi}_0^{(1)} q(u_0^{a_0^1} \mid x_0^{a_0^1}) \Big(\prod_{t=0}^{T-1} \pi(x_{t+1}^{(1)} | x_t^{a_t^1}) q({u_{t+1}}^{a_{t+1}^1} \mid x_{t+1}^{a_{t+1}^1}) \Big) dx_{0:T}^1 du_{0:T}^1 \\
=& Z,
\end{aligned}
$$

where in the second equality, we marginalize the variables not involved in the particle $x_{0:T}^1$ and $x_{0:T}^{a_{0:T}^1}$. The final equality follows because we are averaging over N^T possible cases of $a_{0:T}^1$ and all are equal to Z, which concludes the proof.

D Comparison to Twisted Diffusion Sampler

In this section, we discuss similarities and differences between RDSMC and Twisted Diffusion Sampler [TDS, 30], which is an SMC method for conditional generation from diffusion models.

Structurally, our method shares similarities with TDS, as both methods use some form of "look-ahead" or "twisting" functions [19, Chapter 3] to improve the efficiency of SMC. These twisting functions introduce future information to intermediate states, promote promising particles through the weighting procedure, and improve the SMC sampling efficiency.

Moreover, the incremental weight expression of both TDS and RDSMC takes the same form (their Eq. (11) v.s. our Eq. (15)):

$$w_t := \frac{\psi(x_t)\gamma_t^0(x_{t:T})}{\psi(x_{t+1})\gamma_t^0(x_{t+1:T})q(x_t \mid x_{t+1})},$$

where ψ is the twisting function, γ_t^0 is the untwisted intermediate target and q is the proposal. The product $\gamma_t(x_{t:T}) \coloneqq \psi(x_t)\gamma_t^0(x_{t:T})$ defines a series of *twisted* intermediate targets.

While the expressions of these components are method-specific, the structure of the weighting functions is shared between TDS our RDSMC and follows from the generic twisted SMC framework [19, Chapter 3].

Importantly, this weight construction introduces twisting functions in a telescoping manner such that their approximation errors cancel out over the full trajectory. This property ensures that the final target is correct, and the resulting SMC procedure is asymptotically exact for both TDS and RDSMC.

However, we want to highlight several important *differences* between TDS and RDSMC:

1. Problem setting: The problem we address is fundamentally different from that of TDS. TDS is designed to sample from the conditional distribution $p_\theta(x_0 \mid y) \propto p_\theta(x_0)p(y \mid x_0)$ given a pretrained diffusion model $p_\theta(x_0)$ and a likelihood model $p(y \mid x_0)$ with the observation y. As a comparison, our goal is to sample from an arbitrary distribution $\pi(x_0)$, assuming access to its unnormalized density $\tilde{\pi}(x_0)$.
2. Design of key components: Due to the difference in settings, our intermediate target distributions, proposal distributions, and twisting functions require different designs. In particular,
   - TDS's twisting function is $p(y \mid \hat{x}_\theta(x_t)$ (their Eq. (8)), the likelihood evaluated at the denoising prediction $\hat{x}_\theta(x_t)$ by the pretrained model, which approximates the intractable likelihood $p_\theta(y \mid x_t)$. This twisting function incorporates the observation y into the weighting function (their Eq. (11)). Notably, when $t = 0$, TDS's twisting function recovers the exact likelihood $p(y \mid x_0)$ and no approximation is involved.
     Given the twisting function, TDS's intermediate target is $p\left(y \mid \hat{x}_\theta(x_t)\right) p_\theta(x_{t:T})$ where $p_\theta(x_{t:T})$ is the distribution under the pretrained diffusion model. The proposal is obtained by adding a term that is the gradient of the twisting function, $\nabla_{x_t} \log p(y \mid \hat{x}_\theta(x_t))$, to the pretrained diffusion model's transition kernel (see their Eq (9) and (11)).
   - In our case, RDSMC uses the marginal estimate $\hat{\pi}(x_t, u_t)$ (Eq. (10)) as the twisting function, which approximates the exact marginal $\pi(x_t)$. The marginal integrates out the future information in $\pi(x_0)$, thereby encoding the target information into the intermediate weight.
     We define intermediate targets as $\hat{\pi}(x_t, u_t)\pi(x_{t:T})q(u_{t:T} \mid x_{t:T})$ (a restatement of Eq. (11)); the product term $\hat{\pi}(x_t, u_t)\pi(x_{t:T})$ is structurally similar to TDS's intermediate target, while we additionally account for MC estimation randomness $q(u_{t:T} \mid x_{t:T})$. We also utilize twisting function to inform the proposal, that is, we use the score estimate $s(x_t, u_t)$, which is connected to the twisting function / marginal estimate $\hat{\pi}(x_t, u_t)$, to design the proposal in Eq. (8).
3. Use of auxiliary variables: Our method introduces auxiliary randomness through the Monte Carlo estimation of the twisting function. Incorporating this randomness places RDSMC in the category of nested SMC methods [23]. In contrast, TDS does not use auxiliary variables and operates more like a standard SMC algorithm, without requiring nested sampling.

E Experiment details

This section provides experimental details and additional results.

Appendix E.1 describes the implementation and hyperparameter settings for RDSMC and all baseline methods. Appendices E.2 to E.4 present supplementary information and additional results for the target distributions used in the *main* experiments. Appendix E.5 reports *additional* experiments that control for hyperparameters, runtime, and other relevant factors to compare across methods, as well as an empirical comparison to Phillips et al. [11].

Compute resources. All experiments were conducted on a single NVIDIA A100 GPU with 40 GB GPU memory, and accessed between 16 GB and 32 GB of CPU memory, depending on the task. RDSMC and baseline methods are all implemented in PyTorch [47].

E.1 Method and hyperparameter details

We first give an overview of baseline methods we benchmark RDSMC against in Appendix E.1.1. We then describe key hyperparameter settings of our method and competing baselines for the main experiments in Appendix E.1.2.

E.1.1 Summary of baseline methods

AIS [25]. The annealed importance sampling (AIS) algorithm defines a sequence of geometrically annealed distributions $\rho_t(x)$ for $t \in \{0, 1, \cdots, T\}$ from an initial proposal $\rho_0(x)$ to the desired target $\rho_T(x) \propto \pi(x)$ as $\rho_t(x) \propto \rho_0(x)^{1-\beta_t}\pi(x)^{\beta_t}$ where $\{\beta_t\}_{t=0}^T$ is an increasing linear schedule with each $\beta_t = t/T$. AIS begins by sampling from the initial proposal $\rho_0(x)$ and proceeds through a sequence of MCMC transitions, each targeting the intermediate distribution $\rho_t(x)$. The resulting weighted samples provide a consistent approximation to the target distribution $\pi(x)$. The complexity of AIS is $\mathcal{O}(NTn)$ for N final samples, T annealing steps, and n MCMC transitions per step.

SMC [26]. Sequential Monte Carlo (SMC) operates on the same sequence of annealed distributions as AIS but differs by performing resampling at certain iterations. In our implementation, we consider both constant resampling at each step and an adaptive strategy based on an ESS threshold κ_{ess}. The complexity of SMC is the same as that of AIS.

RDMC [8]. Reverse diffusion Monte Carlo (RDMC) builds on the time-reversed Ornstein–Uhlenbeck diffusion:

$$dY_t = \big\{Y_t + 2\,\nabla \log p_{t_{\text{init}}-t}(Y_t)\big\}\,dt + \sqrt{2}\,dB_t, \quad Y_0 \sim \pi_{t_{\text{init}}}$$

where $p_s(y) = \int \mathcal{N}(y;\, \mathrm{e}^{-s}x,\, (1-\mathrm{e}^{-2s})I)\,\pi(dx)$. By Tweedie's formula, the intractable score $\nabla \log p_{t_{\text{init}}-t}(Y_t)$ can be written in terms of the posterior denoiser:

$$u_t(y) \;=\; \int x\, q_t(x \mid y)\, dx, \qquad q_t(x \mid y) \propto \pi(x)\,\mathcal{N}\big(x;\, \mathrm{e}^{t_{\text{init}}-t}y,\, (\mathrm{e}^{2(t_{\text{init}}-t)}-1)\,I\big),$$

so that the SDE becomes

$$dY_t = \Big\{\tfrac{\mathrm{e}^{-2\,(t_{\text{init}}-t)}+1}{\mathrm{e}^{-2\,(t_{\text{init}}-t)}-1}\,Y_t + \tfrac{2\,\mathrm{e}^{-(t_{\text{init}}-t)}}{1-\mathrm{e}^{-2\,(t_{\text{init}}-t)}}\,u_t(Y_t)\Big\}dt + \sqrt{2}\,dB_t.$$

Following the implementation of Grenioux et al. [9], RDMC discretizes the interval $[0, t_{\text{init}}]$ into T steps. At initialization, RDMC employs a *Langevin-within-Langevin* scheme to sample from $\pi_{t_{\text{init}}}$: using ULA where the score function is estimated based on DSI. In subsequent steps, RDMC simulates from the reverse SDE where the denoiser $u_t(y)$ is obtained by running MALA targeting $q_t(x \mid y)$.

The overall complexity of RDMC is $\mathcal{O}(NM(n_{\text{init}} + Tn))$ where N is the number of final samples, T is the number of iterative sampling steps, and the remaining factors are associated with score estimation: M is the number of MCMC chains, while n_{init} and n denote the number of MCMC transition steps in the initial and subsequent stages, respectively.

SLIPS [9]. Stochastic Localization via Iterative Posterior Sampling (SLIPS) relies on a stochastic observation process defined as:

$$Y_t = \alpha(t)X + \sigma W_t,$$

where $(W_t)_{t\geq 0}$ is a standard Brownian motion independent of $X \sim \pi$, and $\alpha(t)$ is a flexible denoising schedule function. To bypass the direct sampling requirement from π, SLIPS introduces a conditional denoiser defined by:

$$u_t(y) = \int x q_t(x \mid y)dx, \quad q_t(x \mid y) \propto \pi(x)\mathcal{N}\left(\frac{y}{\alpha(t)}, \frac{\sigma^2}{g(t)^2}I\right).$$

The corresponding SDE governing this observation process is:

$$dY_t = \dot{\alpha}(t)u_t(Y_t)dt + \sigma dB_t, \tag{37}$$

where $(B_t)_{t\geq 0}$ is another standard Brownian motion.

Similar to RDMC, the SLIPS algorithm approximates the dynamics defined by this SDE using an MCMC approach to estimate the conditional denoiser u_t or a related score function:

- The SLIPS initialization also involves a *Langevin-within-Langevin* procedure. Different from RDMC, SLIPS chooses the initialization time t_{init} such that both the marginal $\pi_{t_{\text{init}}}(y)$ and the posterior $q_{t_{\text{init}}}(x \mid y)$ are (approximately) log-concave , which they refer to as the "duality of log-concavity" assumption.
- In the subsequent steps, SLIPS integrates the observation process where the denoiser $u_t(Y_t)$ is estimated by MALA sampling from $q_t(x \mid y)$.

Similar to RDMC, SLIPS' complexity is $\mathcal{O}(NM(n_{\text{init}} + Tn))$ where N is the number of final samples, T is the number of iterative sampling steps, and the remaining factors are associated with score estimation: M is the number of MCMC chains, while n_{init} and n denote the number of MCMC transition steps in the initial and subsequent stages, respectively.

SMS [33]. For a target π, Sequential Multimeasurement Sampler (SMS) draws M independent noisy observations at some noise scale $\sigma > 0$:

$$Y^m = X + \sigma Z^m, \quad Z^m \sim \mathcal{N}(0, I), \quad m = 1, \ldots, M,$$

with $X \sim \pi$. For any $m \in \{1, ..., M\}$, the posterior density of X given $y_{1:m}$ is

$$q_m(x \mid y_{1:m}) \;\propto\; \pi(x)\,\mathcal{N}\big(\,\bar{y}_{1:m},\, \tfrac{\sigma^2}{m}I\big),$$

where $\bar{y}_{1:m} = (1/m)\sum_{i=1}^m y_i$. Its Bayes estimator

$$u_m(y_{1:m}) \;=\; \mathbb{E}[X \mid Y^{1:m} = y_{1:m}]$$

serves as a non-Markovian denoiser, and one shows that the law of $u_m(Y^{1:m})$ converges to π in W_2 at rate $O(\sigma\sqrt{d/m})$ [33].

To sample from π, one first simulates the entire M-tuple $Y^{1:M}$ and then returns $u_M(Y^{1:M})$. They employ a 'Once-At-A-Time' strategy: draw Y^1 via ULA, then sequentially sample each $Y^m \mid Y^{1:m-1}$ using MCMC targeting the conditional density of Y^m. Under mild assumptions this sequence of targets becomes increasingly log-concave in m, but the initial draw of Y^1 can be as challenging as sampling from π itself when σ is large. Although one can estimate scores via IS or inner-loop posterior MCMC, numerical results show a steep degradation in performance unless σ is carefully tuned—a manifestation of the same "duality of log-concavity" that forces a trade-off between ease of initialization and overall convergence as reported by Grenioux et al. [48].

The time complexity of SMS is $O(NTMn)$ for N final samples, T iterative sampling steps, n MCMC transition steps per sampling step, and M importance samples used for score estimation.

E.1.2 Implementation details and hyperparameter settings for the main experiments

We summarize the hyperparameter values used to tune RDSMC and its variants for the main experiments in Table 3, where these hyperparameters are explained in Appendix B.3.3. The choices of

Target	M	n	δ_{mcmc}	$t_{\text{start-resampling}}$	score est.	mcmc kernel
GMM ($d = 2, 4$)	{10, 100}	{1, 10}	{1.0}	{0.5, 0.8}	DSI	HMC
GMM ($d = 8$ - 64)	{10, 100}	{ 1, 10, 50}	{1.0}	{0.5, 0.8}	DSI	HMC
Rings	{100}	{1, 10}	{0.01, 0.05}	{0.5}	DSI	MALA
Funnel	{100}	{1, 10, 50}	{0.01, 0.05}	{0.5, 0.8}	DSI	MALA
Logistic Reg.	{ 10, 100}	{10, 50}	{0.01, 0.005}	{0.5, 0.8}	TSI	MALA

Table 3: Hyperparameter grid used for tuning RDSMC in the main experiments. We fix the number of sampling steps at $T = 100$, set $l = 1$ MCMC step per AIS transition, and use an ESS threshold of $\kappa_{\text{ess}} = 0.3$. Score estimator and MCMC kernel types are selected via a coarse grid search using validation metrics. RDSMC (IS) and RDSMC (Proposal) use the same grid with $\kappa_{\text{ess}} = 0$, excluding $t_{\text{start-resampling}}$ as it does apply in these settings.

Target	**AIS**	**SMC**		**SMS**	
	$\delta_{\text{mcmc-init}}$	$\delta_{\text{mcmc-init}}$	κ_{ess}	δ_{mcmc}	δ_γ
GMM	{0.1, 0.5, 1.0}	{0.1, 0.5, 1.0}	{0.3, 1.0}	{0.03, 1.0}	{0.0625, 0.05}
Rings	{0.01}	{0.01}	{0.3, 1.0}	{0.03, 1.0}	{0.0625, 0.05}
Funnel	{0.01}	{0.01}	{0.3, 1.0}	{0.03, 1.0}	{0.0625, 0.05}
Logistic Reg.	{0.01}	{0.01}	{0.3, 1.0}	{0.03, 1.0}	{0.0625, 0.05}

Target	**RDMC**	**SLIPS**		
	$\exp\{t_{\text{init}}\}$	t_{final}	ϵ	$\delta_{\text{mcmc-init}}$
GMM	$\{0.95, 0.9, 0.8, 0.7\}$	{150, 300}	{0.03, 0.05, 0.1, 0.2, 0.4}	{0.1, 0.5, 1.0}
Rings	$\{0.95, 0.9, 0.8, 0.7\}$	{150, 300}	{0.1, 0.2, 0.4, 1.0, 1.2}	{1e-5}
Funnel	$\{0.95, 0.9, 0.8, 0.7\}$	{150, 300}	{0.1, 0.2, 0.4, 1.0, 1.2}	{1e-5}
Logistic Reg.	$\{0.95, 0.9, 0.8, 0.7\}$	{150}	{0.03, 0.05, 0.1, 0.2, 0.4}	{1e-5}

Table 4: Hyperparameter grid used for tuning baseline methods in the main experiments. AIS, SMC, RDMC, and SLIPS use $T = 1,000$ sampling steps, and SMS uses $K = 500$ noisy observations.

score estimator and MCMC kernel types are determined by a coarse grid search using validation metrics; We primarily tune $M, n, \delta_{\text{mcmc}}$ and $t_{\text{start-resampling}}$, as described in Table 3.

The hyperparameter search grid for baseline methods is provided in Table 4. We largely follow the implementation of the official codebase of Grenioux et al. [9] (available at `https://github.com/h2o64/slips`). For the bi-modal GMM experiments, we tune key hyperparameters within a similar range to those used for RDSMC. For the Rings, Funnel, and Bayesian logistic regression tasks, we adopt the settings reported by Grenioux et al. [9].

We use the following target variance information to initialize AIS, SMC, and SLIPS. Crucially, RDMC and our method do not make use of this information.

We restate Assumption A0 from Grenioux et al. [9].

Assumption 1. *(Log-concavity outside a compact). There exist $R > 0$ and $\tau > 0$ such that π is the convolution of μ and $\mathcal{N}(0, \tau^2 \mathbf{I}_d)$, where μ is a distribution compactly supported on $\mathbb{B} = \mathbb{B}(\mathbf{m}_\pi, R)$, i.e., $\mu(\mathbb{R}^d \setminus \mathbb{B}) = 0$.*

The values of R and τ that approximately or exactly satisfy this assumption for different targets are provided in the corresponding experiment subsections.

We next discuss the hyperparameter settings for each of the baseline methods.

AIS. AIS uses $T = 1,000$ annealing steps. Each transition step uses an MALA kernel with an adaptive step size, initialized at $\delta_{\text{mcmc-init}}$, to maintain the acceptance ratio at 75%. Additionally, MALA runs 4 parallel chains, each with 32 MCMC steps. The initial proposal $\rho_0(x)$ is set to $\mathcal{N}(x \mid 0, R^2 + \tau^2)$.

The *only* tunable parameter we consider is $\delta_{\text{mcmc-init}}$.

SMC. SMC follows the same configuration as AIS, except it incorporates ESS-based resampling, introducing the normalized ESS threshold κ_{ess} as an additional hyperparameter (alongside $\delta_{\text{mcmc-init}}$).

RDMC. RDMC uses $T = 1,000$ discretization steps. To initialize $\pi_{t_{\text{init}}}$ in RDMC, the Langevin-within-Langevin algorithm is simulated using 16 MCMC steps and 4 MCMC chains. The chains are initialized with an IS approximation of the posterior powered by 128 particles. The initial sample is drawn from $\mathcal{N}(0, (1 - \exp\{-2t_{\text{init}}\})\mathbb{I}_d\})$ and the initial step size is set to $(1 - \exp\{-2t_{\text{init}}\})/2$

Subsequent steps use a MALA kernel for posterior sampling with adaptive step size to maintain the acceptance ratio at 75%, 4 parallel chains and 32 MCMC steps. We drop the first 50 MCMC samples to ignore the warm-up period in the estimation.

The only tunable parameter we consider is t_{init}.

SLIPS. SLIPS follows a similar initialization and posterior sampling scheme as RDMC, also with $T = 1,000$ discretization steps. However, instead of tuning the value of the initial sampling time t_{init}, SLIPS makes use of the target variance information and sets $\pi_{t_{\text{init}}}(x) := \mathcal{N}(x \mid 0, R^2/d + \tau^2)$.

SLIPS also features a few algorithmic subtleties.

- SLIPS uses a stochastic exponential integrator scheme to simulate from their observational SDE, where ours and RDMC consider the Euler Maruyama scheme.
- SLIPS discretizes the SDE using evenly spaced points in the log-SNR space, while our method and RDMC perform discretization in the time domain.
- SLIPS reuses the final MCMC samples from the previous step as the initialization for MCMC at the next step, promoting continuity across iterations.
- At the final step, SLIPS outputs the estimated denoiser as an approximate sample from the target π to align with standard stochastic localization literature. In contrast, our method and RDMC return a random sample generated from the approximated SDE dynamics.

We consider the following hyperparameters: (i) t_{final} the final integration time of Eq. (37), (ii) ϵ the initial time to determine the log-SNR-based discretization schedule, and (iii) $\delta_{\text{mcmc-init}}$, the initial step size for the MALA kernel.

SMS. We set the number of noisy observations M in SMS to $\lfloor T/2 \rfloor$ where $T = 1,000$, and use as many MCMC steps per noise level as SLIPS or RDMC. This choice of K ensures that the computational complexity of SMS is on par with other baseline methods. The Langevin steps are done using ULA as suggested by the authors.

We tune the step size δ_{mcmc} and the friction coefficient δ_γ using recommended values by the authors.

E.2 Bi-moddal gaussian mixturess

Target. We consider a Gaussian mixture model (GMM) with two components in a d-dimensional space. The component means m_1, m_2 are randomly initialized from a uniform distribution over a box of width 80, centered at the origin. Each component has diagonal covariance with all diagonal values equal to $\sigma^2 = 2 \log 2$. The weights are fixed to $[w_1, w_2] = [0.1, 0.9]$ regardless of dimension d. Formally, the target density is

$$\pi(x) = w_1 \mathcal{N}(x \mid m_1, \sigma^2 \mathbb{I}_d) + w_2 \mathcal{N}(x \mid m_2, \sigma^2 \mathbb{I}_d). \tag{38}$$

Metrics. We compute the estimated weights by assigning samples to different components.

For a sample x, we assign it to the component $k \in \{1, 2\}$ with highest posterior probability value, which is computed as

$$p(x \text{ in component } k) = \frac{w_k \, \mathcal{N}(x \mid \mu_k, \sigma^2 \mathbb{I}_d)}{\sum_{j=1}^{2} w_j \, \mathcal{N}(x \mid \mu_j, \sigma^2 \mathbb{I}_d)}.$$

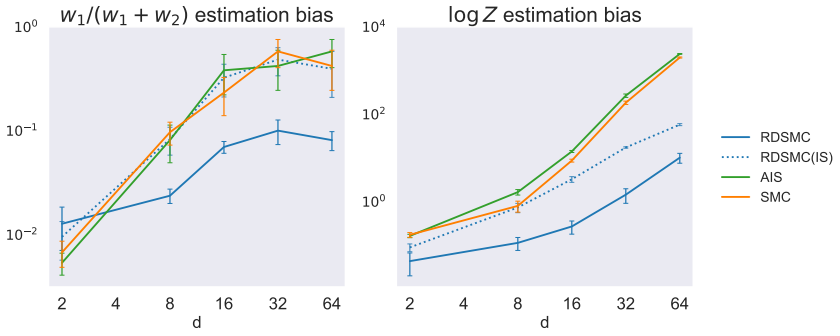

Figure 2: Estimation bias of weight ratio $w_1/(w_1+w_2)$ and log normalization constant $\log Z$, with hyperparameters selected to minimize the $\log Z$ estimation bias. We observe that all methods exhibit monotonic trends in both bias metrics.

The weight estimates are then formulated as $\hat{w}_1 = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}\{x_i \text{ in component } 1\}$ and $\hat{w}_2 = 1 - \hat{w}_1$.

The weight estimation ratio bias is computed as $|\hat{w}_1/(\hat{w}_1+\hat{w}_2) - w_1/(w_1+w_2)| = |\hat{w}_1 - w_1|$.

The log normalization constant estimation bias is simply $|\hat{Z} - Z|$ where $\hat{Z}$ is the estimated value, for method that computes it.

Additional implementation details. Assumption 1 are satisfied for $R = \max(w_1, w_2)\|m_1 - m_2\| = 0.9\|m_1 - m_2\|$, and $\tau = \sigma$. These values are used to initialize AIS, SMC, SMS, and SLIPS.

For RDSMC and its variants, we use DSI to estimate the score. The initial proposal distribution in Algorithm 3 is set to $\mathcal{N}(x_0 \mid x_t/\alpha_t, \sigma_t^2/\alpha_t^2)$.

Additional results. The hyperparameters used for results in Figure 1b are selected based on minimizing the estimation bias of weight ratio $w_1/(w_1+w_2)$. Here, we also report results using hyperparameters tuned by minimizing the $\log Z$ bias in Figure 2. Results for RDSMC (Proposal), SMS, RDMC, and SLIPS are not included as they do not provide normalization constant estimate. The trends are largely similar, except that the $\log Z$ estimation bias curve appears monotonic.

E.3 Rings and Funnel distributions

Targets. The Rings distribution [9] is defined via an inverse polar reparameterization of a base distribution p_z, which is factorized into two independent univariate marginals p_r and p_θ:

- p_r is a mixture of four Gaussian distributions, each with mean located at integer radial positions $i+1$ for $i \in \{0,1,2,3\}$ and variance 0.15^2.
- The angular component p_θ is uniformly distributed over $[0, 2\pi]$.

The Funnel distribution [34] is characterized by a hierarchical density structure in a 10-dimensional space, where the first dimension x_1 is drawn from a Gaussian distribution with zero mean and variance $\sigma^2 = 9$. The subsequent dimensions $x_{2:10}$ are conditionally Gaussian, with variances exponentially dependent on the first dimension. We follow Grenioux et al. [9] to use the following Funnel density

$$\pi(x_{1:10}) = \mathcal{N}(x_1; 0, \sigma^2)\mathcal{N}(x_{2:10}; 0, \exp(x_1)\mathbb{I}_9). \tag{39}$$

Metrics. The entropic regularized 2-Wasserstein distance between two distributions μ and ν is defined by

$$W_{2,\varepsilon}(\mu,\nu) = \inf\{\textstyle\int_{\mathbb{R}^d\times\mathbb{R}^d} \|x_1 - x_0\|^2 \mathrm{d}\pi(x_0,x_1) - H(\pi) : \pi_0 = \mu, \pi_1 = \nu\}^{1/2}, \tag{40}$$

where $\varepsilon > 0$ is a regularization hyper-parameter and $H(\pi) = -\int_{\mathbb{R}^d\times\mathbb{R}^d} \log \pi(x_0,x_1)\mathrm{d}\pi(x_0,x_1)$ refers to as the entropy of π. We evaluate the quality of sampling with regularization $\varepsilon = 0.05$ via POT library [49].

Total Variation Distance (TVD) measures the discrepancy between the true and model-generated distributions by comparing their marginal histograms within a bounded region.

Let $\hat{\mu}_i$ and $\hat{\nu}_i$ represent the probability mass in the i-th bin for the true and generated distributions, respectively. We compute the TVD as follows

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

For the radius TVD, we generate the histogram of the radius samples using 256 uniform bins over the interval $[0, 8]$. For the angle TVD, we generate the histogram of angle samples using 256 uniform bins over the interval $[-\pi, \pi]$.

The Kolmogorov-Smirnov distance (KSD) between two distributions μ and ν is defined by

$$\text{KSD}(\mu, \nu) = \sup_{x \in \mathbb{R}^d} \|F_\mu(x) - F_\nu(x)\| , \tag{41}$$

where F_μ and F_ν denotes the cumulative distribution function of μ and ν respectively. We evaluate the 10-dimensional Funnel samples using the sliced version from [48, Appendix D.1].

Additional implementation details. For both targets, we use DSI to estimate the score. On Rings, we choose $R = 4, \tau = 0.15$ that satisfy Assumption 1 to initialize AIS, SMC, SMS, and SLIPS; and on Funnel, we use $R = 2.12, \tau = 0.0$, following the implementation of Grenioux et al. [9].

On Rings, we set the initial proposal distribution in Algorithm 3 for RDSMC and its variants to $\mathcal{N}(x_0 \mid x_t/\alpha_t, \sigma_t^2/\alpha_t^2)$. On Funnel, the above choice would render numerical instability when t is large. Hence we use $\mathcal{N}(x_0 \mid x_t, \sigma_t^2/\alpha_t^2)$ as the proposal for the AIS procedure.

E.4 Bayesian logistic regression

Targets. We consider four Bayesian logistic regression datasets: Credit and Cancer [35], Ionosphere [36] and Sonar [37].

- Credit [35]: this dataset addresses binary classification of individuals as good or bad credit risks. It contains 1000 data points with 25 standardized features.
- Cancer [35]: this dataset involves classifying recurrence events in breast cancer. It consists of 569 data points in a 31-dimensional standardized feature space.
- Ionosphere [36]: this dataset classifies radar signals passing through the ionosphere into good or bad categories. It features 351 data points with dimensionality $d = 35$.
- Sonar [37]: this dataset differentiates sonar signals reflected from metal cylinders versus cylindrical rocks. It includes 208 data points with $d = 61$ standardized features.

We consider a training dataset $\mathcal{D} = \{(x_j, y_j)\}_{j=1}^M$ where $x_j \in \mathbb{R}^d$ and $y_j \in \{0, 1\}$ for all $j \in \{1, \ldots, M\}$. A Bayesian logistic regression model consists of the following

$$p(w, b) = \mathcal{N}(w; 0, \mathbf{I}_d)\mathcal{N}(b; 0, (2.5)^2) \tag{42}$$

$$p(y \mid x; w, b) = \text{Bernoulli}(y; \sigma(x^\top w + b)) \tag{43}$$

where $w \in \mathbb{R}^d$ is a weight vector, $b \in \mathbb{R}$ is an intercept, and σ is the sigmoid function.

The target distribution we are interested in is the posterior distribution

$$p(w, b \mid \mathcal{D}) \propto p(\mathcal{D} \mid w, b)p(w, b) = \prod_{j=1}^M p(y_j \mid x_j; w, b)p(w, b). \tag{44}$$

Metric. Following Grenioux et al. [9], we use the following definition of the predictive log-likelihood given a test dataset $\mathcal{D}_{\text{test}}$:

$$\frac{1}{N} \sum_{i=1}^N \sum_{(x,y) \in \mathcal{D}_{\text{test}}} \log p(w_i, b_i) + \log p(y \mid x; w_i, b_i) \tag{45}$$

where $\{w_i, b_i\}_{i=1}^N$ are posterior samples.

Additional implementation details. On all tasks, we use $R = 2.5, \tau = 0.0$ in Assumption 1 to initialize AIS, SMC, SMS, and SLIPS following the implementation of Grenioux et al. [9].

For RDSMC and its variants, we use TSI to estimate the score and clip the norm of the estimated score within 20 to improve numerical stability. We use $\mathcal{N}(x_0 \mid x_t/\alpha_t, \sigma_t^2/\alpha_t^2)$ as the initial proposal distribution in the AIS-based score estimation procedure.

E.5 Ablation experiments

In our main experiments, baseline methods follow the implementation of Grenioux et al. [9] which use more discretization steps T compared to our method, and result in different runtimes. Additionally, AIS, SMC and SLIPS use target variance information while RDSMC does not.

In this section, we conduct ablation studies to control for these factors and evaluate performance under comparable computational budgets. We discuss the hyperparameter settings in Appendix E.5.1. We present the results for the targets in the main experiments with the new hyperparameter settings in Appendix E.5.2.

In addition, we include an empirical comparison to the Particle Denoising Diffusion Sampler [11] in Appendix E.5.3.

E.5.1 Hyperparameter settings

To enable fair comparisons under similar computational budgets, we fix the compute budget for RDSMC and align the hyperparameters of SLIPS, RDMC, and SMS accordingly, since these methods have comparable computational complexity; however, algorithmic differences lead to variations in runtime. For AIS and SMC, we adjust their hyperparameters to achieve total runtimes comparable to that of RDSMC.

In addition, only SLIPS and SMS use target scalar variance to initialize their samplers, while other methods including RDSMC do not access this information.

We summarize hyperparameter settings for each method below.

RDSMC.

- Use $T = 100$ uniform discretization timesteps
- Use AIS score estimator based on the DSI, and MALA as the MCMC kernel in AIS
- Use an adaptive MCMC step size, similar to that used in the baseline SLIPS, AIS, and SMC methods (see description in Appendix B.3.3). This adaptive strategy differs from the fixed step size employed in the main experiments which requires tuning. The initial MCMC step size is 1 for GMM targets and 0.05 for others.
- Use $M = 100$ AIS importance samples
- Use $n = 50$ AIS annealing steps
- Use $m = 1$ MCMC transition per AIS step
- Use $\kappa_{\text{ess}} = 0.3$ for the ESS-based resampling threshold
- The *only* tunable hyperparameter is the starting step for resampling, $t_{\text{start-resampling}}$, which we vary in $\{0, 0.1, 0.2, \cdots, 1.0\}$ unless otherwise specified. Recall that we apply resampling at step t when $t \leq t_{\text{start-resampling}}$ and the normalized ESS $< \kappa_{\text{ess}}$.

We treat RDSMC (IS) as a special case of RDSMC with $t_{\text{start-resampling}} = 0$, which is included within the generic RDSMC framework as part of hyperparameter tuning. The final hyperparameter values tuned by validation metrics are reported in Table 5.

RDSMC (Proposal) uses the same configuration as RDSMC except that no resampling or weighting is applied throughout the sampling process.

AIS and SMC. To generate N final samples, recall that the computational complexity of AIS and SMC is $\mathcal{O}(NTn)$ for T annealing steps and n MCMC transitions per step; in contrast, RDSMC's

Target	start resampling time ($t_{\text{start-resampling}}$)
GMM ($d = 2$)	0.2
GMM ($d = 4$)	0.6
GMM ($d = 8$)	0.4
GMM ($d = 16$)	0.2
GMM ($d = 32$)	0.1
GMM ($d = 64$)	0.4
Rings	0.1
Funnel	1.0
Credit	0.9
Cancer	0.6
Ionosphere	0.7
Sonar	1.0

Table 5: Ablation experiments: final hyperparameter settings for RDSMC, selected by target-dependent validation metrics.

Target	# of annealing steps (T)	# of MCMC transitions per step (n)
GMM ($d = 2, 4, 8$)	100	100
GMM ($d = 16$)	150	100
GMM ($d = 32$)	240	100
GMM ($d = 64$)	400	100
Rings	100	70
Funnel	100	100
Credit	1000	140
Cancer	1000	90
Ionosphere	1000	70
Sonar	1000	60

Table 6: Ablation experiments: final hyperparameter settings for AIS and SMC, tuned to achieve comparable runtime to RDSMC (with $T = 100$ time steps, $M = 100$ AIS importance samples, and $n = 50$ AIS annealing steps) on a single NVIDIA A100 GPU (40 GB memory). While the M importance samples in RDSMC can be processed in parallel, computation can still result in slower performance on a single device. We therefore increase T and n for AIS and SMC to match the runtime of RDSMC.

complexity is $\mathcal{O}(NTMn)$ for T discretization steps, M AIS importance samples, n AIS annealing steps (with a minor abuse of notation).

While the M AIS importance samples in RDSMC can be processed in parallel, computation can still result in in slower runtime on a single device. Therefore, we adjust the number of annealing steps T and the number of MCMC transitions per step n to ensure that the total runtime on a single NVIDIA A100 GPU (40 GB memory) is comparable to that of RDSMC. The resulting hyperparameter values are provided in Table 6.

Additionally, in contrast to the main experiments following Grenioux et al. [9], we do *not* use the target variance to initialize the initial proposal in AIS and SMC. Instead, we use the initial proposal $\mathcal{N}(0, 1)$ which coincides with the base distribution of RDSMC. We use the same adaptive MCMC step size strategy as in RDSMC. For SMC, the ESS-based resampling threshold is set to 0.3.

Other aspects of AIS and SMC are the same as in the main experiments.

SLIPS and RDMC. We match the computational budget of each score estimation step to that of RDSMC. Specifically, both SLIPS and RDMC use $T = 100$ discretization steps; in every step, they use $M = 100$ parallel MCMC chain, each with $n = 50$ transitions, for score estimation. Consequently, the asymptotic complexity of SLIPS and RDMC (after the initialization stage) is the same as that of RDSMC, which is $\mathcal{O}(NTMn)$ to produce N final samples.

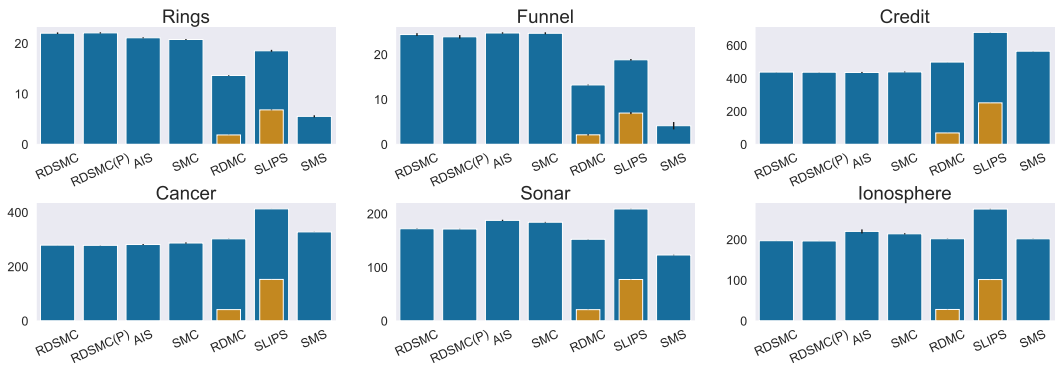

Figure 3: Ablation experiments: runtimes (in seconds) for all methods across different targets. Each blue bar represents the total wall-clock time averaged over 10 random seeds with error bar denoting 2 standard errors, measured on a single NVIDIA A100 GPU (40 GB). For RDMC and SLIPS, we also include the initialization time indicated by the orange bars. RDSMC(P) denotes RDSMC (Proposal). The GMM results are reported separately in the final panel of Figure 4. AIS and SMC are configured to achieve runtimes comparable to RDSMC. While RDMC, SLIPS and SMS have the same asymptotic complexity of RDSMC, their actual runtimes differ due to algorithmic differences.

However, because SLIPS and RDMC require additional computation for initialization and their score estimation relies on MCMC, whereas RDSMC starts from the base distribution and uses AIS for score estimation, the overall runtimes of these methods differ.

Other aspects of SLIPS and RDMC are the same as in the main experiments

SMS. We use $T = 100$ steps, $n = 50$ MCMC transitions per sampling step and $M = 100$ importance samples for score estimation. Other aspects of SMS are the same as in the main experiments.

E.5.2 Ablation experiment results

We revisit the targets studied in the main experiment using the new hyperparameter settings and summarize the results. Each experiment is run with 10 random seeds, and hyperparameters are selected using the corresponding validation metrics.

Runtime. The overall runtimes for all methods across all targets are shown in Figure 3, except for the GMM targets, which are presented in the final panel of Figure 4.

We note that AIS and SMC are manually configured to achieve runtimes comparable to RDSMC. While RDMC, SLIPS and SMS have the same asymptotic complexity of RDSMC, their actual runtimes differ due to algorithmic differences. In GMM (with varying d), Rings, and Funnel, we observe shorter runtimes for RDMC, SLIPS, and SMS compared to RDSMC. In contrast, for Credit, Cancer, Sonar, and Ionosphere, SLIPS tends to have a longer runtime. Notably, the initialization phase of RDMC and SLIPS accounts for a substantial portion of their total runtime.

Bi-modal GMM. We report the results in Figure 4. In the first two panels, we observe that for both the weight ratio $w_1/(w_1 + w_2)$ and the log normalization constant $\log Z$, RDSMC consistently outperforms the other methods. These results are consistent with the findings in the main experiments in Figure 1, despite a different hyperparameter setting. The last panel summarizes the runtime comparison.

The optimal hyperparameters for each method are selected based on the lowest estimation bias of the weight ratio, consistent with the procedure used in the main experiments.

Rings and Funnel. We report the results in Table 7. On synthetic targets, AIS and SMC achieve the lowest average Radius TVD and $\log Z$ bias on Rings and lowest average $\log Z$ bias on Funnel, while RDSMC achieves comparable results on these metrics and the lowest average Sliced KSD on Funnel.

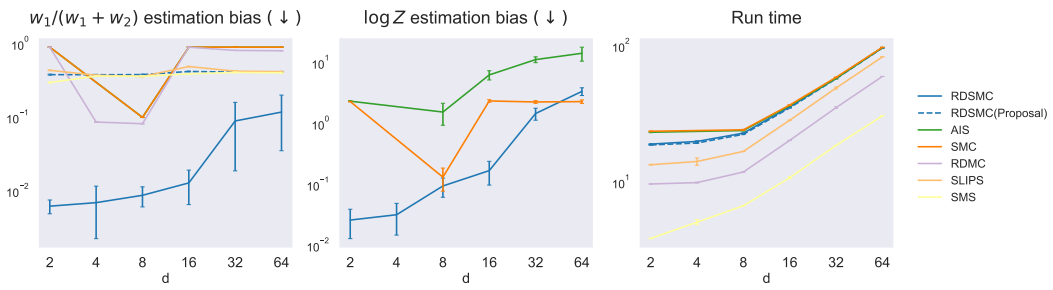

Figure 4: Ablation experiments: estimation bias of weight ratio $w_1/(w_1 + w_2)$ (left) and log-normalization constant $\log Z$ (middle), and runtime (in seconds, right) versus dimension d. Results are averaged over 10 seeds with error bars showing one standard error. Note that SMS, RDMC, and SLIPS do not provide estimates of $\log Z$. For both metrics, the estimation bias increases with d for all methods, while RDSMC consistently outperforms the baselines. The runtime for RDSMC (including the Proposal variant). SMC and AIS are configured for comparable runtime of RDSMC, whereas SLIPS, RDMC, and SMS exhibit shorter runtimes owing to algorithmic differences, despite having the same asymptotic complexity as RDSMC.

Algorithm	Rings		Funnel	
	Radius TVD ↓	log Z bias ↓	Sliced KSD ↓	log Z bias ↓
RDSMC	0.118 ± 0.002	$\mathbf{0.013 \pm 0.004}$	$\mathbf{0.075 \pm 0.005}$*	$\mathbf{0.286 \pm 0.068}$
RDSMC(Proposal)	$\mathbf{0.091 \pm 0.003}$	N/A	0.327 ± 0.033	N/A
RDMC	0.440 ± 0.003	N/A	0.129 ± 0.000	N/A
SLIPS	0.279 ± 0.003	N/A	$\mathbf{0.076 \pm 0.001}$	N/A
AIS	$\mathbf{0.088 \pm 0.003}$	$\mathbf{0.004 \pm 0.002}$*	$\mathbf{0.087 \pm 0.002}$	$\mathbf{0.173 \pm 0.010}$
SMC	$\mathbf{0.088 \pm 0.002}$*	$\mathbf{0.006 \pm 0.002}$	$\mathbf{0.082 \pm 0.004}$	$\mathbf{0.169 \pm 0.012}$*
SMS	0.282 ± 0.010	N/A	0.161 ± 0.000	N/A

Table 7: Ablation experiments: Rings and Funnel targets (mean $\pm$ standard error over 10 seeds). Bold indicates 95% confidence interval overlap with that of the best average result. AIS and SMC achieve the lowest average Radius TVD and $\log Z$ bias on Rings, and the lowest $\log Z$ bias on Funnel, with RDSMC remaining comparable when accounting for standard errors. RDSMC achieves the lowest Sliced KSD on Funnel.

Note that these observations differ slightly from those in the main experiments (Table 1) due to changes in the hyperparameter settings. In particular, AIS and SMC no longer use target-variance scaling for initialization, which appears to improve their performance in this case.

In this new setting, AIS and SMC demonstrate more robust overall performance. While RDSMC remains competitive, it exhibits greater variability in the reported metrics (indicated by the larget standard errors). Nevertheless, RDSMC tends to outperform diffusion-based MC samplers that do not have SMC correction, including its Proposal variant, SLIPS, and RDMC.

The optimal hyperparameters for each method are selected based on the lowest Radius TVD for Rings and lowest Sliced KSD for Funnel, consistent with the procedure used in the main experiments.

Finally, we note that for RDSMC on the Rings target, when the hyperparameter $t_{\text{start-resampling}} > 0.8$, the magnitude of the estimated $\log Z$ becomes significantly larger than when $t_{\text{start-resampling}} \leq 0.8$. For this reason, we tune $t_{\text{start-resampling}} \in \{0, 0.1, \ldots, 0.8\}$, within which the estimated $\log Z$ values remain consistent.

Bayesian logistic regression. Different from the main experiments in Table 8, we consider a different metric Test LPPD, the log pointwise predictive density on the test set, defined as follows

$$\text{Test LPPD} = \sum_{(x,y)\in\mathcal{D}_{\text{test}}} \log\left(\int p(y_i \mid x_i; w, b)\, p(w, b \mid \mathcal{D})\, \mathrm{d}w, b\right) \tag{46}$$

Test LPPD ↑	Credit ($d = 25$)	Cancer ($d = 31$)	Ionosphere ($d = 35$)	Sonar ($d = 61$)
RDSMC	**-94.54 ± 1.34**	**-10.59 ± 0.45**	**-25.97 ± 0.73**	**-18.54 ± 0.28**
RDSMC(Proposal)	-94.62 ± 0.06	-50.24 ± 0.16	**-24.87 ± 0.02***	-18.91 ± 0.01
RDMC	-138.22 ± 0.35	-78.03 ± 0.12	-44.16 ± 0.07	-28.64 ± 0.03
SLIPS	**-92.42 ± 0.02***	-10.41 ± 0.01	-25.22 ± 0.01	**-18.40 ± 0.01***
AIS	**-92.91 ± 0.44**	**-10.13 ± 0.07**	-25.09 ± 0.02	**-18.41 ± 0.01**
SMC	**-92.55 ± 0.05**	**-10.13 ± 0.01***	-25.09 ± 0.02	**-18.41 ± 0.01**
SMS	-98.00 ± 0.27	-20.47 ± 0.16	-26.17 ± 0.10	-23.29 ± 0.08

Table 8: Ablation experiments: Bayesian logistic regression with test log pointwise predictive density (Test LPPD, with mean ± standard error) averaged over 10 seeds. Bold indicates 95% confidence interval overlap with that of the best average result. RDSMC, AIS, SMC, and SLIPS achieve the best overall performance. RDSMC outperforms RDSMC (Proposal) and RDMC in most cases, highlighting the effectiveness of the SMC correction.

where the likelihood $p(y_i \mid x_i; w, b)$ is defined in Eq. (43) and the posterior $p(w, b \mid \mathcal{D})$ is defined in Eq. (44).

We use final (weighted) samples from each method to approximate the Test LPPD in Eq. (46).

We report the results in Table 8. We observe that RDSMC, AIS, SMC, and SLIPS achieve the best overall performance. Compared to SLIPS, RDSMC does not require target-variance-based initialization. Moreover, RDSMC outperforms RDSMC (Proposal) and RDMC in most cases, highlighting the effectiveness of the SMC correction. However, we observe that results for RDSMC exhibit large variance, as the case in the main experiments from Table 2.

The optimal hyperparameters for each method are selected based on the highest LPPD on a heldout validation set, computed analogously to the Test LPPD.

E.5.3 Comparison to Particle Denoising Diffusion Sampler on the Funnel target

Finally, we present a comparison to the Particle Denoising Diffusion Sampler (PDDS) [11], which is also a diffusion-based SMC sampler. However, unlike RDSMC, PDDS requires training.

We evaluate PDDS on the Funnel target using 10 random seeds. Experiments are conducted on an NVIDIA RTX A6000 GPU, whereas our previous experiments use an NVIDIA A100 GPU. In addition, PDDS is implemented in JAX (following the official implementation at `https://github.com/angusphillips/particle_denoising_diffusion_sampler#`), while RDSMC and the other baselines are implemented in PyTorch.

PDDS requires approximately 82.63 ± 0.43 seconds for training and 15.08 ± 0.073 seconds for final sampling of 4096 particles, resulting in a total runtime of about 97.70 ± 0.47 seconds (reported as mean ± standard error). The runtimes of our method and other training-free baselines are reported in Figure 3, which are all below 25 seconds. However, due to differences in hardware and software frameworks, PDDS's runtime is not directly comparable to those of the previous experiments.

PDDS achieves a $\log Z$ bias of 0.26 ± 0.04 and a mean Sliced KSD of 0.10 ± 0.00. Compared with the results in Table 7, PDDS exhibits a slightly lower average $\log Z$ bias than RDSMC, but higher than AIS and SMC. Its Sliced KSD is also higher than that of RDSMC, AIS, and SMC.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our method provides a diffusion-based SMC samplers with theoretical guarantees. Our claims match our theory results and empirical results.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss limitations of our work in the Experiment section when comparing to other methods, as well as in Discussion where we look out for future directions.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: We provide full set of assumptions and a complete proof in Appendix C.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide the algorithm in the main text, and include all the hyperparemter settings in Appendix. Our codes are available here `https://github.com/LuhuanWu/RDSMC`. Baseline methods are taken from a publically available codebase `https://github.com/h2o64/slips`.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Our codes are available here `https://github.com/LuhuanWu/RDSMC`.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We include all experiment details in the main text and appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report standard errors in tables and figures, and highlight results that are within 2 standard errors of the best one.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We include details in computing resources in appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethnics and the research conduced in the paper conforms with it.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: This paper studies the sampling problem which is ubiquitous in statistical inference and various applications. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper does not pose such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We include these details in Appendix.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: LLM is used only for editing purposes and is not involved in any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.