# AMORTIZING TEMPERED SAMPLING WITH CORRECTED SELF-TRAINING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Learning a neural sampler can reduce the cost of repeated sampling from an unnormalized distribution, but obtaining good training data remains difficult. We develop a *corrected self-training* scheme across a sequence of tempered targets, making use of importance sampling and Markov chain Monte Carlo to correct the samples and replay buffers to store them. We introduce Sequential Score Sampler (SSS) as a particular case. In SSS, the proposal is a score-based diffusion trained by regression on energy gradients; and the samples are corrected by importance weights on the path space. After training, samples can be collected directly from the model, from its importance-corrected proposals, or from the final-temperature buffer. The method amortizes repeated sampling by encoding the global structure of the target in the model. SSS is competitive in terms of mean discrepancy with the two baselines we considered, while taking less wall-clock time.

## 1 INTRODUCTION

Many inference and simulation problems require repeated sampling from a distribution whose density is known only up to a normalizing constant. Multimodality makes this difficult: local Markov chain Monte Carlo (MCMC) may explore a basin thoroughly while rarely moving between basins (Woodard et al., 2009). Annealed importance sampling and sequential Monte Carlo (SMC) address this difficulty through intermediate distributions (Neal, 2001; Chopin, 2002; Del Moral et al., 2006; Dai et al., 2022). However, obtaining further accurate estimates still requires additional sampling computation. Can some of that computation instead be invested in learning a reusable proposal?

Learned samplers pursue this amortization (Zhang & Chen, 2022; Vargas et al., 2023; Midgley et al., 2023; Akhound-Sadegh et al., 2024). Once trained, a generative model can generate new proposal samples without continuing a long local Markov chain. The difficulty is how to train it without representative target samples. One approach is to generate samples from the current model, correct for proposal mismatch, and use the resulting data to update it. Importance weighting supplies this correction by adjusting the samples' contributions without requiring the target's normalizing constant.

We develop corrected self-training along a sequence of tempered targets and instantiate it with Sequential Score Sampler (SSS). Rather than fit a separate proposal at each temperature, we train one model that takes the temperature as an input. Initial sample populations can come from a suitable sampler or existing problem-specific simulations. During training, the model generates proposals at each intermediate temperature. We resample them according to their importance weights and apply MCMC steps that preserve the corresponding target. Reweighting to the next temperature then provides data for its buffer. These buffers retain weighted samples from recent iterations, allowing to reuse costly sampling results over multiple optimization steps instead of training only on the latest batch. Each outer iteration updates the model using data from all temperatures.

In SMC with local Markov transitions, reweighting and resampling cannot create configurations outside the current population; reaching an unrepresented basin therefore depends on rare transitions. A learned global proposal can instead reuse information encoded from earlier samples and across temperatures (Dibak et al., 2022), without starting from a surviving particle. Problem-specific architecture can support this reuse. For interacting-particle systems, for example, an equivariant network shares learned geometric relations across rotations and permutations of identical particles (Satorras et al., 2021). Such structure reduces the need to learn equivalent configurations separately and can support proposals beyond those stored in a finite population.



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

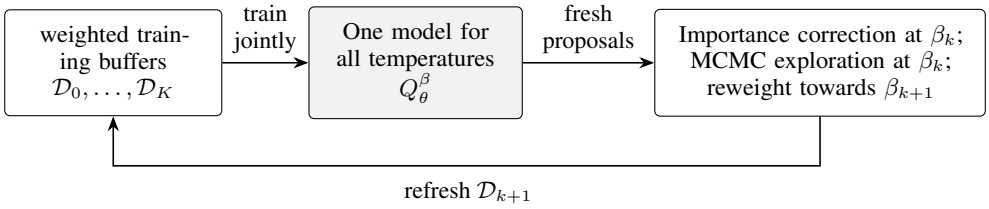



Figure 1: Corrected self-training across temperatures.

Section 2 presents the general training scheme, independently of the proposal family and initialization sampler. Section 3 gives the score-based implementation. In Section 4, we compare raw model samples, importance-corrected samples, and final-buffer samples against Temperature-Annealed Boltzmann Generators (TA-BG; Schopmans & Friederich, 2025) and Progressive Inference-Time Annealing (PITA; Akhound-Sadegh et al., 2025).

## 2 Corrected self-training across temperatures

### 2.1 Targets and initialization

Let $\rho$ be a normalized reference density from which sampling is easy and whose support covers the target. We write the target as $\pi(x) = Z_1^{-1}\rho(x)e^{-E(x)}$ and define the intermediate unnormalized and normalized densities

$$\bar{\pi}_\beta(x) = \rho(x)e^{-\beta E(x)}, \qquad \pi_\beta(x) = \frac{\bar{\pi}_\beta(x)}{Z_\beta}, \qquad 0 \le \beta \le 1. \tag{1}$$

Here $Z_\beta = \int \bar{\pi}_\beta(x)\,\mathrm{d}x$, so $\pi_0 = \rho$ and $\pi_1 = \pi$. We call $\beta$ the 'temperature' parameter. Choose a ladder $0 = \beta_0 < \cdots < \beta_K = 1$ and obtain initial weighted populations approximating the targets at its levels. These choices may use problem knowledge, existing simulation data, or a preliminary sampler such as SMC or parallel tempering. Adjacent targets should overlap sufficiently for incremental importance reweighting. Our adaptive-SMC initialization is one particular implementation of this step.

Let $Q_\theta^\beta$ denote the distribution generated by the learned proposal when given temperature $\beta$. The same parameters $\theta$ are shared across all levels; in a neural sampler, they are the network weights. We write $q_\theta(x \mid \beta)$ for its density.

For each target $\pi_{\beta_k}$, we maintain a weighted training dataset $\mathcal{D}_k$, called a replay buffer, initially filled with the corresponding population. Retaining data across refreshes is useful when generating corrected samples is expensive relative to a model update: each sample can be reused in several optimization steps, and training need not rely on a single, potentially noisy batch. Pooling recent batches also retains configurations from earlier iterations while the proposal changes.

Training alternates between refreshing these datasets with corrected proposals and fitting the shared model to data from all temperatures. A refresh at $\beta_k$ supplies data for $\mathcal{D}_{k+1}$. Figure 1 summarizes the data flow, and Algorithm 1 specifies the update order.

### 2.2 Generating corrected training data

For each selected level $k \in \{1, \ldots, K-1\}$, we refresh $\mathcal{D}_{k+1}$ using corrected proposals from the preceding, typically easier target $\pi_{\beta_k}$. Well-initialized buffers may be kept fixed; our choice of refreshed levels is specified in Section 4.1.

**Algorithm 1** Corrected self-training across temperatures

**Input:** Reference $\rho$, energy $E$, ladder $\{\beta_k\}_{k=0}^{K}$, initialized weighted buffers $\{\mathcal{D}_k\}_{k=0}^{K}$

```
Initialize the shared model parameters θ.
for each outer iteration do
    for each selected level k in parallel do
        Generate N proposals and importance weights at β_k using Q_θ^{β_k}
        Resample by these weights; apply π_{β_k}-invariant MCMC updates.
        Compute v_i = π̄_{β_{k+1}}(x_i)/π̄_{β_k}(x_i).
        Update D_{k+1} with this weighted batch (Section 2.2.3).
    Update θ using weighted minibatches from all buffers.
return The trained proposal, its importance-weight calculation, and D_K.
```

#### 2.2.1 Generating weighted proposals at $\beta_k$

At fixed model parameters $\theta$, generate $N$ proposal samples $x_i$ and nonnegative importance weights $w_i$ targeting $\pi_{\beta_k}$. We require the importance-sampling consistency property: for test functions $f$,

$$\widehat{\mu}_{\beta_k,N}(f) = \frac{\sum_{i=1}^{N} w_i f(x_i)}{\sum_{i=1}^{N} w_i} \longrightarrow \mathbb{E}_{\pi_{\beta_k}}[f] \qquad \text{as } N \to \infty. \tag{2}$$

When the proposal density is available, the weight is simply $w_i = \bar{\pi}_{\beta_k}(x_i)/q_\theta(x_i \mid \beta_k)$, with the usual support condition. Otherwise, importance sampling can be performed on an extended space that includes the sampler's auxiliary variables, provided the extended target has marginal $\pi_{\beta_k}$. Section 3.2 gives such a construction for diffusion paths.

#### 2.2.2 Resampling and applying MCMC

We resample the proposals according to their importance weights and apply $\pi_{\beta_k}$-invariant MCMC updates. The weights adjust the proposals' contributions to the target approximation. The Markov moves can diversify duplicated particles and explore nearby configurations while preserving the target distribution.

#### 2.2.3 Reweighting and updating the buffer at $\beta_{k+1}$

To turn the resulting population into training data for the next target, assign each sample the incremental importance weight

$$v_{k\to k+1}(x) = \frac{\bar{\pi}_{\beta_{k+1}}(x)}{\bar{\pi}_{\beta_k}(x)} = \exp\{-(\beta_{k+1} - \beta_k)E(x)\}\,. \tag{3}$$

Unlike the proposal-correction weights $w_i$, these weights account only for the change of target from $\beta_k$ to $\beta_{k+1}$. We insert this weighted batch into $\mathcal{D}_{k+1}$. In our implementation, each buffer has fixed capacity and uses first-in, first-out (FIFO) replacement: new batches gradually replace the oldest entries. This retains data across several refreshes without keeping all past populations indefinitely. Before insertion, we normalize weights within each incoming batch to have mean one, giving equal-sized batches equal total weight. Appendix A.2 gives the details.

### 2.3 Learning one model from all temperatures

After refreshing the selected buffers, we draw training samples from every $\mathcal{D}_k$ with probabilities proportional to their stored weights. Each temperature contributes the same number of samples to a training minibatch. The fitting objective depends on the proposal family: a normalizing flow can be trained by weighted maximum likelihood, whereas SSS uses the score-regression loss in Section 3.1. Gradients are taken only through this fitting objective, not through sample generation, importance weights, resampling, or MCMC. Sharing parameters allows features learned at one temperature to influence proposals at others, while retaining training data from the entire ladder throughout optimization.

### 2.4 FINAL SAMPLE COLLECTION

After training, the construction supports three ways to collect samples at the target temperature, evaluated in Section 4 as SSS (raw), SSS (corrected), and SSS (buffer).

**Raw model samples.** Draw directly from $Q_\theta^1$, without importance correction or MCMC. The sampling cost is that of the proposal itself, but its distribution may differ from the target.

**Importance-corrected samples.** Generate proposals at $\beta = 1$ together with their importance weights. These can be used in weighted estimates or to resample the proposals; we evaluate the latter option. The weighted estimates satisfy (2), but poor proposal-target overlap can concentrate the weights on a few samples, leading to low effective sample size and many duplicates after resampling.

**Final-buffer samples.** Draw from $\mathcal{D}_K$ according to its stored weights. This population has already undergone the correction, MCMC exploration, and temperature-bridging steps during training. Its pool of distinct configurations is limited by the buffer capacity; obtaining new configurations requires further buffer updates.

## 3 A SCORE-BASED IMPLEMENTATION

This section describes a particular instantiation of the general preceding construction. SSS uses a score-based diffusion proposal and the standard Gaussian reference $\rho = \mathcal{N}(0, I_d)$. The temperature $\beta$ identifies the target, whereas diffusion time $t$ controls Gaussian smoothing within that target.

### 3.1 LEARNING THE RELATIVE SCORE

For $X \sim \pi_\beta$, define $X_t = \alpha_t X + \sqrt{1-\alpha_t^2}\,\varepsilon$, where $\alpha_t = e^{-t}$ and $\varepsilon \sim \mathcal{N}(0, I_d)$ is independent of $X$. Let $\pi_{\beta,t}$ denote the density of $X_t$. Its score relative to the Gaussian reference is

$$r^*(z, t, \beta) = \nabla_z \log \frac{\pi_{\beta,t}(z)}{\rho(z)} = -\alpha_t \beta\, \mathbb{E}[\nabla E(X) \mid X_t = z]. \tag{4}$$

The Gaussian identity in the second equality is derived in Appendix B and underlies relative-score and target-score regression (Phillips et al., 2024; Bortoli et al., 2024). Since the reference score $-z$ is known, the network need only learn the deviation from it. We approximate $r^*$ with a neural network $r_\theta$ by minimizing

$$\mathcal{L}(\theta) = \mathbb{E}\Big[\|r_\theta(X_t, t, \beta_k) + \alpha_t \beta_k \nabla E(X)\|^2\Big]. \tag{5}$$

The expectation draws $k$ uniformly from $\{0, \dots, K\}$, then $X$ from $\mathcal{D}_k$ with probability proportional to its stored weight. Independently, it draws $t \sim \mathrm{Unif}[t_{\min}, T]$ and $\varepsilon \sim \mathcal{N}(0, I_d)$ to construct $X_t$ as above. Uniform selection of $k$ gives equal weight to every temperature.

For exact target samples, least-squares regression recovers the conditional expectation in (4). Buffer samples approximate this population objective, so missing regions or incorrect relative mode masses can bias the learned score. Refreshing the buffers combines importance correction with target-invariant MCMC exploration to improve the fidelity and coverage of the training data. Energy gradients are required only at the unperturbed buffer samples and can be cached and reused across diffusion times and optimization steps. At $\beta_0 = 0$, the relative score is zero at all times; we initialize the network's output near zero.

### 3.2 GENERATING SAMPLES AND COMPUTING WEIGHTS

The learned relative score defines the reverse-time diffusion

$$Y_0 \sim \rho, \qquad \mathrm{d}Y_s = \big[-Y_s + 2r_\theta(Y_s, T-s, \beta)\big]\,\mathrm{d}s + \sqrt{2}\,\mathrm{d}W_s, \quad 0 \le s \le T. \tag{6}$$

With an accurate relative score and a horizon $T$ for which $\pi_{\beta,T}$ is close to $\rho$, its terminal distribution approximates $\pi_\beta$. We use an exponential integrator that treats the Gaussian-reference dynamics exactly. Appendix A.3 gives the numerical update and its implementation. The numerical sampler produces a path $y_{0:L}$ starting at $y_0 \sim \rho$, with endpoint $x = y_L$. For $h = T/L$ and $a = e^{-h}$, its Gaussian transition density is $q_{\theta,n}(\cdot \mid y, \beta) = \mathcal{N}\big(ay + 2(1-a)r_\theta(y, T-nh, \beta),\, (1-a^2)I_d\big)$.

Let $g_h(y' \mid y) = \mathcal{N}(y'; ay, (1-a^2)I_d)$ be the exact Ornstein–Uhlenbeck reference transition. A chain started at $\rho$ and propagated with $g_h$ still has endpoint distribution $\rho$. Multiplying its path density by $e^{-\beta E(y_L)}$ therefore gives an extended target with endpoint marginal $\bar{\pi}_\beta$. The ratio of this unnormalized path density to the proposal path density is

$$w(y_{0:L}) = e^{-\beta E(y_L)} \prod_{n=0}^{L-1} \frac{g_h(y_{n+1} \mid y_n)}{q_{\theta,n}(y_{n+1} \mid y_n, \beta)}. \tag{7}$$

For fixed $\theta$ and integrable $f$, this construction gives $\mathbb{E}[w(y_{0:L}) f(y_L)] = Z_\beta \mathbb{E}_{\pi_\beta}[f]$, where the first expectation is over proposal paths. Independent simulations therefore satisfy (2). The correction applies to the discrete sampler actually used: it requires neither an exact score nor evaluation of the endpoint density $q_\theta(x \mid \beta)$. The weights can be accumulated online without storing the path (Appendix A.3).

# 4 Experiments

## 4.1 Targets, network architectures, and baselines

**Targets.** We consider three families of targets with different structures. A factorized Gaussian mixture in $d$ dimensions has $2^d$ well-separated modes and allows exact sampling of ground-truth data. We run the experiments for two cases, $d = 16$ and $d = 50$. Comparing the two settings shows how the method scales with dimension while the per-coordinate structure of the target is fixed. The four-particle double-well system (DW-4) in two dimensions tests multimodality in the configurations of interacting particles. The thirteen-particle Lennard-Jones system (LJ-13) in three dimensions adds a stiff repulsive potential and harmonic confinement. DW-4 and LJ-13 follow the benchmarks of Köhler et al. (2020). Appendix C.1 provides their exact parameters needed for reproduction.

**Initialization and rejuvenation.** In all experiments, one adaptive SMC run selects the temperature ladder and initializes the buffers (Del Moral et al., 2006; Jasra et al., 2011; Beskos et al., 2016). We use preconditioned Hamiltonian Monte Carlo (HMC) for local rejuvenation; its tuning and the reuse of the SMC-tuned parameters are specified in Appendix A.1. The reference buffer $\mathcal{D}_0$ contains independent $\rho$-samples. The first bridge weights such samples by $e^{-\beta_1 E(x)}$, followed by SMC resampling and Markov moves, to initialize $\mathcal{D}_1$. We keep $\mathcal{D}_0$ and $\mathcal{D}_1$ fixed: the former already contains exact reference draws, and the latter provides an SMC-based training population at the first nontrivial level without relying on a learned proposal. Only $\mathcal{D}_2, \ldots, \mathcal{D}_K$ are refreshed, using levels $k = 1, \ldots, K-1$ in Algorithm 1.

**Network architecture.** In all experiments, a single network $r_\theta(x_t, t, \beta)$ is shared across all temperatures of the ladder, modeling the relative score $r^*(x_t, t, \beta)$ described in Section 3.1. We use a residual multilayer perceptron (MLP) for the Gaussian mixtures, and an E($n$)-equivariant graph neural network (EGNN) (Satorras et al., 2021) for DW-4 and LJ-13. Appendix C.2 gives the details.

**Baselines.** We compare SSS with two recent samplers that also train a generative model on self-generated data across a temperature ladder. The first is TA-BG (Schopmans & Friederich, 2025). TA-BG trains a normalizing flow by reverse KL at a high temperature, where barriers are low enough to avoid mode collapse, and then anneals it to the target temperature. At each iteration, it importance-reweights flow samples to the next temperature, resamples them, and trains the flow by forward KL on the resampled buffer. The second is PITA (Akhound-Sadegh et al., 2025). PITA trains a sequence of diffusion models from high to low temperature and, alongside each score model, fits an energy-based model of the diffusion marginals. This model defines a Feynman-Kac PDE whose weights, accumulated along the reverse SDE and corrected by SMC resampling, produce training samples at the next lower temperature. PITA thus spreads annealing over diffusion time instead of applying it only at the endpoint.

Both baselines share the overall structure of SSS but differ in what is learned and where the correction is applied. TA-BG reweights only at the endpoint and needs a tractable likelihood. PITA reweights along diffusion time but must train an auxiliary energy model, its main stated limitation. SSS needs neither. Its path-space weights in (7) come from the proposal's own Gaussian transition kernels,

Table 1: Sampler performance across four targets, reported as Distance-$\mathcal{W}_2$, Energy-$\mathcal{W}_2$, and Geometric-$\mathcal{W}_2$; lower is better. Entries are mean $\pm$ standard deviation over 3 seeds. For each target and metric, **bold** marks the best entry and blue the second best, as determined by Welch's two-sample $t$-test at $p < 0.1$. —means not applicable and $^*$ means value above $10^4$ and not meaningful. See Appendix C for details.

| TARGET | METHOD | Distance-$\mathcal{W}_2 \downarrow$ | Energy-$\mathcal{W}_2 \downarrow$ | Geometric-$\mathcal{W}_2 \downarrow$ |
|---|---|---|---|---|
| GMM16 ($d = 16$) | TA-BG | $0.04 \pm 0.03$ | $0.20 \pm 0.04$ | — |
| | PITA | $4.18 \pm 4.91$ | $795.69 \pm 1340.00$ | — |
| | SSS (raw) | $0.08 \pm 0.03$ | $1.22 \pm 0.23$ | — |
| | SSS (corrected) | $\mathbf{0.01 \pm 0.01}$ | $0.06 \pm 0.02$ | — |
| | SSS (buffer) | $0.03 \pm 0.01$ | $\mathbf{0.03 \pm 0.00}$ | — |
| GMM50 ($d = 50$) | TA-BG | $1.30 \pm 0.23$ | $95.92 \pm 14.19$ | — |
| | PITA | $2.93 \pm 1.04$ | $517.97 \pm 171.27$ | — |
| | SSS (raw) | $\mathbf{0.14 \pm 0.04}$ | $5.38 \pm 1.05$ | — |
| | SSS (corrected) | $0.20 \pm 0.17$ | $2.73 \pm 3.35$ | — |
| | SSS (buffer) | $0.17 \pm 0.12$ | $\mathbf{0.82 \pm 0.62}$ | — |
| DW-4 ($d = 8$) | TA-BG | $0.38 \pm 0.03$ | $83.09 \pm 110.41$ | $0.90 \pm 0.03$ |
| | PITA | $0.54 \pm 0.07$ | $1.14 \pm 0.04$ | $1.08 \pm 0.13$ |
| | SSS (raw) | $0.22 \pm 0.08$ | $2.68 \pm 0.81$ | $0.52 \pm 0.15$ |
| | SSS (corrected) | $0.19 \pm 0.02$ | $0.15 \pm 0.04$ | $0.49 \pm 0.04$ |
| | SSS (buffer) | $\mathbf{0.18 \pm 0.02}$ | $\mathbf{0.10 \pm 0.02}$ | $\mathbf{0.45 \pm 0.03}$ |
| LJ-13 ($d = 39$) | TA-BG | $1.21 \pm 0.02$ | $61.47 \pm 0.12$ | $4.16 \pm 0.01$ |
| | PITA | $0.04 \pm 0.00$ | $2.26 \pm 0.21$ | $1.65 \pm 0.00$ |
| | SSS (raw) | $0.10 \pm 0.04$ | $^*$ | $1.65 \pm 0.09$ |
| | SSS (corrected) | $0.16 \pm 0.10$ | $13.76 \pm 6.44$ | $2.41 \pm 0.25$ |
| | SSS (buffer) | $\mathbf{0.02 \pm 0.01}$ | $\mathbf{0.76 \pm 0.05}$ | $\mathbf{1.48 \pm 0.00}$ |

Table 2: Wall-clock time (in minutes) for each method to produce the reported results. All runs were executed on a single NVIDIA RTX 4500 Ada Generation GPU. The LJ-13 results of TA-BG and PITA are taken from Akhound-Sadegh et al. (2025), so no timing is reported.

| TARGET | TA-BG | PITA | SSS |
|---|---|---|---|
| GMM16 ($d = 16$) | 10 | 70 | 6 |
| GMM50 ($d = 50$) | 83 | 188 | 43 |
| DW-4 ($d = 8$) | 25 | 87 | 15 |
| LJ-13 ($d = 39$) | — | — | 272 |

and a single network is shared across the whole ladder instead of being fine-tuned temperature by temperature.

**Cost and evaluation.** Total cost includes initialization, training, and deployment. Wall-clock time is reported in minutes. Accuracy is assessed with three metrics (Appendix C.4). Distance-$\mathcal{W}_2$ is the one-dimensional 2-Wasserstein distance between pooled summary statistics: coordinates for the Gaussian mixtures and pairwise interparticle distances for DW-4 and LJ-13. Energy-$\mathcal{W}_2$ is the one-dimensional 2-Wasserstein distance between the empirical energy distributions. Because pooled statistics and energies do not determine the full configuration distribution, for DW-4 and LJ-13 we also report Geometric-$\mathcal{W}_2$, the 2-Wasserstein distance between whole configurations under a ground distance that aligns out particle permutations, rotations, and reflections.

### 4.2 MAIN RESULTS

Table 1 reports the three 2-Wasserstein metrics for all methods on the four targets. We reimplement TA-BG and PITA from their published code and evaluate the final checkpoints on an NVIDIA RTX 4500 Ada Generation GPU, with one exception. On LJ-13, training of both baselines diverged on this hardware, so we report the results published by Akhound-Sadegh et al. (2025), which were obtained with a larger computational budget. Appendix C.5 gives detailed training configurations for the

reproduction of all reported results. The three SSS rows are obtained from the same trained model. Across all targets and metrics, the SSS buffer consistently ranks first or second. Except for LJ-13, the best two entries are always SSS sample sets.

**Gaussian mixtures.** On GMM16, SSS and TA-BG perform well, while PITA is unstable across seeds. The SSS corrected samples obtain the best Distance-$\mathcal{W}_2$ and the SSS buffer samples have the best Energy-$\mathcal{W}_2$. On GMM50, which has $2^{50}$ modes, both baselines degrade sharply; for example, the Energy-$\mathcal{W}_2$ of TA-BG rises from $0.20$ to $95.92$. In contrast, all three SSS sample sets retain good sample quality. The raw samples have the lowest mean Distance-$\mathcal{W}_2$, indicating good agreement in the pooled coordinate distribution, but not establishing correct joint mode occupancy. Their larger Energy-$\mathcal{W}_2$ shows residual energy-distribution error, which is smaller for the corrected samples and smallest for the buffer. The greater variability of the corrected samples across seeds is consistent with concentrated importance weights.

**Particle systems.** On DW-4, all three SSS sample sets outperform both baselines on Distance-$\mathcal{W}_2$ and Geometric-$\mathcal{W}_2$, and the buffer is the best on all three metrics. Importance correction reduces the Energy-$\mathcal{W}_2$ of the raw samples from $2.68$ to $0.15$, below that of PITA. TA-BG shows a large mean energy error with substantial variation across runs. LJ-13 is the most challenging target due to its stiff short-range repulsion. On our hardware, training of TA-BG and PITA diverged, whereas SSS trained stably. Compared with the published PITA results, obtained with a larger budget, the SSS buffer is more accurate on all three metrics. The raw samples match the reported mean Geometric-$\mathcal{W}_2$ of PITA, indicating comparable error under this geometric comparison despite their large energy discrepancy. Correction is less effective here since concentrated weights reduce sample diversity. The buffer has lower errors on all three metrics, consistent with the benefit of retaining corrected populations and exploring locally with HMC.

**Choice of sample set.** The three sample sets offer different trade-offs. Raw samples require only network evaluations, but agreement in coordinate or geometric statistics can coexist with substantial energy errors. Corrected samples improve several discrepancies on GMM16 and DW-4, but can lose diversity when the weights concentrate. The buffer combines diffusion proposals, importance correction, and HMC exploration. This makes SSS buffer samples nearly always show the best results in every reported comparison. However, the generated sample size is limited by the buffer capacity.

**Computational cost.** Table 2 reports the wall-clock time each method requires to converge on each target. For our method, it includes SMC initialization, training, and sampling. We run every method until its performance no longer improves. Every reported time corresponds to the accuracy reported in Table 1; stopping earlier reduces the cost but degrades the results. SSS converges fastest on all targets where the baselines could be trained. It is about $1.7$-$1.9\times$ faster than TA-BG and $4$-$12\times$ faster than PITA, while reaching a comparable or higher accuracy.

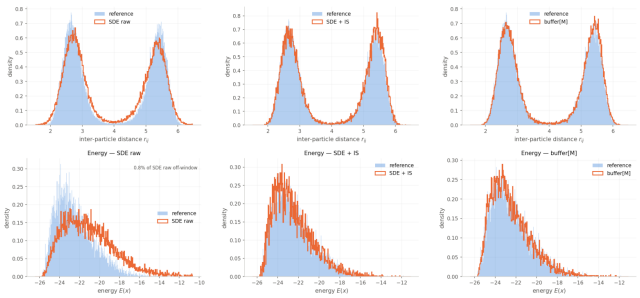
### 4.3 Ablation experiments

We now investigate different aspects of SSS through a set of ablation studies. We run these experiments on DW-4 to examine the roles of importance-correction and MCMC rejuvenation. Specifically, we address two questions.

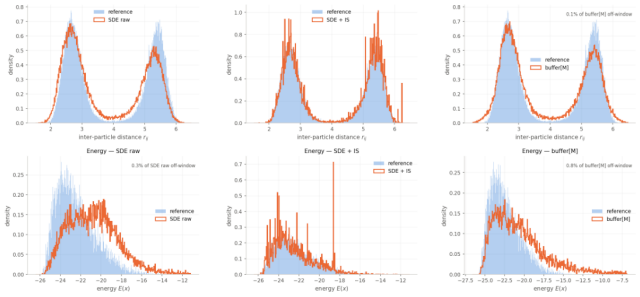
**Question 1 (Path-space weight resampling).** The diffusion can propose samples in different modes without assigning them the correct relative masses. Does importance resampling improve these masses, or are the retained MCMC updates sufficient?

**Question 2 (MCMC rejuvenation).** After resampling, MCMC explores nearby configurations and diversify duplicated particles. How does this exploration affect the training data and subsequent learning?

To answer Question 1, we compare SSS with a variant that omits the path-space importance correction: it uses the same architecture, SMC initialization, HMC budget, temperature bridge, and training schedule. Only the resampling by the correction weights is removed when refreshing the buffers. Figure 2 compares the two runs on DW-4. Without path-space correction, both modes of the pairwise-distance distribution remain visible, but their relative masses are distorted. The same distortion



(a) SSS full method.



(b) SSS without path-space correction.

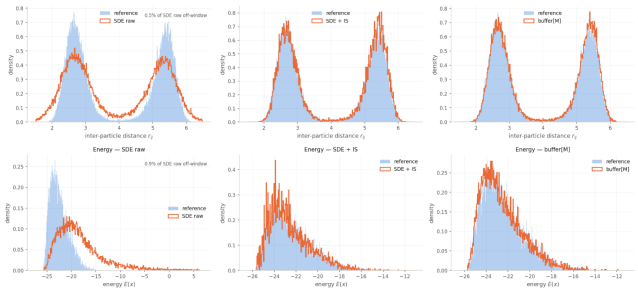
Figure 2: Ablation of the path-space correction on DW-4, final generated samples (orange) and ground truth (blue). Each panel shows the three sample sets of SSS (from left to right): raw samples, importance-corrected samples, and the replay buffer. The top row shows the distribution of inter-particle distances $r_{ij}$ and the bottom row shows the energy distribution.

appears in the final buffer, and the buffer also acquires a long tail of high-energy configurations. These batches are then reused for training. The normalized path-space ESS, computed from the weights in (7) at $\beta = 1$, falls from $0.41$ to $0.06$. Thus, we conclude that the importance-correction, and not the MCMC updates, is what keeps the training data faithful to the relative masses of the target.

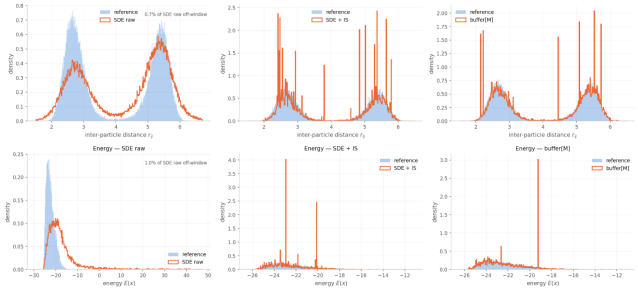
To answer Question 2, we compare SSS with a variant that omits the HMC steps. Both runs share the architecture, SMC initialization, path-space correction mechanism, temperature bridge, and training schedule. Only the rejuvenation step is removed when refreshing the buffers. Figure 3 compares the two runs at the same intermediate training iteration. Without rejuvenation, the resampled batches consist of a few configurations repeated many times, which appear as sharp spikes in the distance and energy histograms. These spikes are inherited by the training buffer, so the model is trained on a small number of distinct configurations. Consequently, training is slower and less stable. Specifically, at the recorded iteration, the raw samples of the full method already reproduce both distance modes and a plausible energy distribution, whereas those of the variant remain far from the reference and carry a long tail of high-energy configurations. The normalized path-space ESS at $\beta = 1$ reflects the same gap, dropping by almost two orders of magnitude, from $0.236$ to $0.003$, when rejuvenation is removed. These results indicate that MCMC exploration improves the diversity of the importance-resampled training populations and accelerates learning in this experiment.

## 5 Related work

**Learning proposals with Monte Carlo.** Annealed Flow Transport and CRAFT combine learned maps with sequential weighting and MCMC (Arbel et al., 2021; de G. Matthews et al., 2022); FAB learns a reusable flow from annealed-importance-sampling computations (Midgley et al., 2023). Reinforced SMC couples learned proposals and twist functions with SMC, off-policy reinforcement learning, and replay (Choi et al., 2025). SCLD likewise combines controlled diffusions with sequential resampling (Chen et al., 2025). Other methods also combine amortization and Monte Carlo correction (Doucet et al., 2022; Vargas et al., 2024; Albergo & Vanden-Eijnden, 2025). Our approach trains one



(a) SSS full method.



(b) SSS without HMC rejuvenation.

Figure 3: Ablation of the HMC rejuvenation on DW-4, evaluated at $\beta = 1$ midway through training (orange) and ground truth (blue). Each panel shows the three sample sets of SSS (from left to right): raw samples, importance-corrected samples, and the replay buffer. The top row shows the distribution of inter-particle distances $r_{ij}$ and the bottom row the energy distribution.

sampler with temperature as an input, using supervised learning on repeatedly refreshed, corrected training data from all levels.

**Learning across temperatures.** PTSD trains diffusion samplers sequentially across temperatures using parallel tempering (Swendsen & Wang, 1986; Geyer, 1991; Hukushima & Nemoto, 1996; Earl & Deem, 2005) and MCMC refinement (Rissanen et al., 2025). TA-BG learns a flow through temperature reweighting (Schopmans & Friederich, 2025). PITA combines progressive diffusion training with inference-time annealing to produce lower-temperature data (Akhound-Sadegh et al., 2025). We share their use of easier targets to train harder ones, an idea also exploited by temperature-conditioned generative models (Dibak et al., 2022; Wang et al., 2022; Invernizzi et al., 2022), but jointly train a shared model across all temperatures.

**Score learning and weighted stochastic samplers.** iDEM alternates generation and energy-based score learning (Akhound-Sadegh et al., 2024), while importance-weighted score matching corrects the distribution used in a score-matching objective (Wang et al., 2025). Our relative-score target uses known Gaussian identities (Efron, 2011; Vincent, 2011; Phillips et al., 2024; Bortoli et al., 2024). Other weighting construction in the literature include Boltzmann generators combining learned equilibrium proposals with reweighting (Noé et al., 2019; Klein et al., 2023; Tan et al., 2025), and stochastic normalizing flows computing trajectory weights without marginalizing stochastic transitions (Wu et al., 2020), building on nonequilibrium path-space reweighting (Jarzynski, 1997; Crooks, 1999; Del Moral et al., 2006; Nilmeier et al., 2011).

## AI Use Statement

In this work, we used generative AI tools to assist in the writing of proofs, design or provide feedback on research methodology or experiments, implement methods, clean and reformat dataset, and support qualitative and thematic data analysis. We have not used generative AI tools to generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims,

provide critical ingredients for proving mathematical claims, propose or refine hypotheses, assist with translation, or interpret results. We have reviewed all AI-assisted work. All AI-outputted texts and AI-suggested edits were read and checked by the authors. Any AI-assisted code was tested by the authors before use. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

## REPRODUCIBILITY STATEMENT

The method is specified by Algorithms 1-2 and Equations (5)-(7), with algorithmic details in Appendix A and a justification of the score formula in Appendix B. Appendix C gives the target definitions, network architectures, evaluation metrics, numerical settings, and the description on re-implementation of the baselines. All results are averaged over three seeds and were computed on a single NVIDIA RTX 4500 Ada Generation GPU, except for the LJ-13 baseline results, which are taken from Akhound-Sadegh et al. (2025). Code to reproduce our method is available at https://anonymous.4open.science/r/progressive_is_tempering-1E43.

## REFERENCES

Tara Akhound-Sadegh, Jarrid Rector-Brooks, Avishek Joey Bose, Sarthak Mittal, Pablo Lemos, Cheng-Hao Liu, Marcin Sendera, Siamak Ravanbakhsh, Gauthier Gidel, Yoshua Bengio, Nikolay Malkin, and Alexander Tong. Iterated denoising energy matching for sampling from Boltzmann densities. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pp. 760–786. PMLR / OpenReview.net, 2024.

Tara Akhound-Sadegh, Jungyoon Lee, Joey Bose, Valentin De Bortoli, Arnaud Doucet, Michael M. Bronstein, Dominique Beaini, Siamak Ravanbakhsh, Kirill Neklyudov, and Alexander Tong. Progressive inference-time annealing of diffusion models for sampling from boltzmann densities. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025.

Michael Samuel Albergo and Eric Vanden-Eijnden. NETS: A non-equilibrium transport sampler. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025.

Michael Arbel, Alexander G. de G. Matthews, and Arnaud Doucet. Annealed flow transport Monte Carlo. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 318–330. PMLR, 2021.

Alexandros Beskos, Ajay Jasra, Nikolas Kantas, and Alexandre Thiery. On the convergence of adaptive sequential Monte Carlo methods. *Ann. Appl. Probab.*, 26(2):1111–1146, 2016. ISSN 1050-5164. doi: 10.1214/15-AAP1113.

Valentin De Bortoli, Michael J. Hutchinson, Peter Wirnsberger, and Arnaud Doucet. Target score matching. *CoRR*, abs/2402.08667, 2024. doi: 10.48550/ARXIV.2402.08667. URL https://doi.org/10.48550/arXiv.2402.08667.

Junhua Chen, Lorenz Richter, Julius Berner, Denis Blessing, Gerhard Neumann, and Anima Anandkumar. Sequential controlled Langevin diffusions. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025.

Sanghyeok Choi, Sarthak Mittal, Víctor Elvira, Jinkyoo Park, and Nikolay Malkin. Reinforced sequential Monte Carlo for amortised sampling. *CoRR*, abs/2510.11711, 2025. doi: 10.48550/ARXIV.2510.11711. URL https://doi.org/10.48550/arXiv.2510.11711.

Nicolas Chopin. A sequential particle filter method for static models. *Biometrika*, 89(3):539–551, 2002. ISSN 0006-3444. doi: 10.1093/biomet/89.3.539.

Gavin E Crooks. Entropy production fluctuation theorem and the nonequilibrium work relation for free energy differences. *Physical Review E*, 60(3):2721, 1999.

Chenguang Dai, Jeremy Heng, Pierre E. Jacob, and Nick Whiteley. An invitation to sequential Monte Carlo samplers. *J. Am. Stat. Assoc.*, 117(539):1587–1600, 2022. ISSN 0162-1459. doi: 10.1080/01621459.2022.2087659.

Alexander G. de G. Matthews, Michael Arbel, Danilo Jimenez Rezende, and Arnaud Doucet. Continual repeated annealed flow transport Monte Carlo. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 15196–15219. PMLR, 2022.

Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. Sequential Monte Carlo samplers. *J. R. Stat. Soc., Ser. B, Stat. Methodol.*, 68(3):411–436, 2006. ISSN 1369-7412. doi: 10.1111/j.1467-9868.2006.00553.x.

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

Arnaud Doucet, Will Grathwohl, Alexander G. de G. Matthews, and Heiko Strathmann. Score-based diffusion meets annealed importance sampling. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

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

Bradley Efron. Tweedie's formula and selection bias. *J. Am. Stat. Assoc.*, 106(496):1602–1614, 2011. ISSN 0162-1459. doi: 10.1198/jasa.2011.tm11181.

Charles J. Geyer. Markov chain Monte Carlo maximum likelihood. In *Computing Science and Statistics: Proceedings of the 23rd Symposium on the Interface*, pp. 156–163, 1991.

Walter R. Gilks and Carlo Berzuini. Following a moving target – Monte Carlo inference for dynamic Bayesian models. *J. R. Stat. Soc., Ser. B, Stat. Methodol.*, 63(1):127–146, 2001. ISSN 1369-7412. doi: 10.1111/1467-9868.00280.

Koji Hukushima and Koji Nemoto. Exchange Monte Carlo method and application to spin glass simulations. *Journal of the Physical Society of Japan*, 65(6):1604–1608, 1996.

Michele Invernizzi, Andreas Kramer, Cecilia Clementi, and Frank Noé. Skipping the replica exchange ladder with normalizing flows. *The Journal of Physical Chemistry Letters*, 13(50):11643–11649, 2022.

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

Ajay Jasra, David A. Stephens, Arnaud Doucet, and Theodoros Tsagaris. Inference for Lévy-driven stochastic volatility models via adaptive sequential Monte Carlo. *Scand. J. Stat.*, 38(1):1–22, 2011. ISSN 0303-6898. doi: 10.1111/j.1467-9469.2010.00723.x.

Leon Klein, Andreas Krämer, and Frank Noé. Equivariant flow matching. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

Jonas Köhler, Leon Klein, and Frank Noé. Equivariant flows: Exact likelihood generative learning for symmetric densities. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5361–5370. PMLR, 2020.

Laurence Illing Midgley, Vincent Stimper, Gregor N. C. Simm, Bernhard Schölkopf, and José Miguel Hernández-Lobato. Flow annealed importance sampling bootstrap. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

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

Jerome P Nilmeier, Gavin E Crooks, David DL Minh, and John D Chodera. Nonequilibrium candidate Monte Carlo is an efficient tool for equilibrium simulation. *Proceedings of the National Academy of Sciences*, 108(45):E1009–E1018, 2011.

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

Gabriel Peyré and Marco Cuturi. Computational optimal transport. With applications to data sciences. *Found. Trends Mach. Learn.*, 11(5-6):1–262, 2018. ISSN 1935-8237. doi: 10.1561/2200000073.

Angus Phillips, Hai-Dang Dau, Michael John Hutchinson, Valentin De Bortoli, George Deligiannidis, and Arnaud Doucet. Particle denoising diffusion sampler. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, volume 235 of *Proceedings of Machine Learning Research*, pp. 40688–40724. PMLR / OpenReview.net, 2024.

Severi Rissanen, Ruikang Ouyang, Jiajun He, Wenlin Chen, Markus Heinonen, Arno Solin, and José Miguel Hernández-Lobato. Progressive tempering sampler with diffusion. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025.

Victor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E(n) equivariant graph neural networks. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 9323–9332. PMLR, 2021.

Henrik Schopmans and Pascal Friederich. Temperature-annealed Boltzmann generators. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025. URL https://proceedings.mlr.press/v267/schopmans25a.html.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

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

Charlie B. Tan, Joey Bose, Chen Lin, Leon Klein, Michael M. Bronstein, and Alexander Tong. Scalable equilibrium sampling with sequential Boltzmann generators. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025.

Francisco Vargas, Will Sussman Grathwohl, and Arnaud Doucet. Denoising diffusion samplers. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

Francisco Vargas, Shreyas Padhy, Denis Blessing, and Nikolas Nüsken. Transport meets variational inference: Controlled Monte Carlo diffusions. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

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

Chenguang Wang, Xiaoyu Zhang, Kaiyuan Cui, Weichen Zhao, Yongtao Guan, and Tianshu Yu. Importance weighted score matching for diffusion samplers with enhanced mode coverage. *CoRR*, abs/2505.19431, 2025. doi: 10.48550/ARXIV.2505.19431.

Yihang Wang, Lukas Herron, and Pratyush Tiwary. From data to noise to data for mixing physics across temperatures with generative artificial intelligence. *Proceedings of the National Academy of Sciences*, 119(32):e2203656119, 2022.

Dawn B. Woodard, Scott C. Schmidler, and Mark Huber. Conditions for rapid mixing of parallel and simulated tempering on multimodal distributions. *Ann. Appl. Probab.*, 19(2):617–640, 2009. ISSN 1050-5164. doi: 10.1214/08-AAP555.

Hao Wu, Jonas Köhler, and Frank Noé. Stochastic normalizing flows. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

Qinsheng Zhang and Yongxin Chen. Path integral sampler: A stochastic control approach for sampling. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

# Appendix

## CONTENTS


**A Algorithmic details** **14**

A.1 Adaptive SMC initialization . . . . . 14

A.2 Replay buffers and per-batch weight normalization . . . . . 15

A.3 Numerical generation and stable log weights . . . . . 15

**B Justification of the score formula** **16**

**C Experimental details** **17**

C.1 Target definitions and reference distributions . . . . . 17

C.2 Neural network architecture . . . . . 17

C.3 Projection onto the zero center-of-mass subspace . . . . . 18

C.4 Reported metrics . . . . . 19

C.5 Numerical settings and baselines reimplementation . . . . . 20


## A ALGORITHMIC DETAILS

### A.1 ADAPTIVE SMC INITIALIZATION

Starting from $N_{\mathrm{SMC}}$ independent draws from $\rho$ with uniform weights, adaptive SMC chooses each next temperature so that the incremental importance ESS equals a prescribed fraction $\eta$ of the population size. Specifically, at step $k$, reweighting the equally weighted population $\{x_i^{(k)}\}$ to a candidate $\beta > \beta_k$ gives the incremental log weights $-(\beta - \beta_k)E(x_i^{(k)})$ and the normalized ESS

$$\mathrm{ESS}_k(\beta) = \frac{\left(\sum_i w_i(\beta)\right)^2}{N_{\mathrm{SMC}} \sum_i w_i(\beta)^2}, \qquad w_i(\beta) = \exp\{-(\beta - \beta_k)E(x_i^{(k)})\}. \tag{A.1}$$

If $\mathrm{ESS}_k(1) \geq \eta$, we set $\beta_{k+1} = 1$ and the sequence ends. Otherwise, $\beta_{k+1}$ is the root of $\mathrm{ESS}_k(\beta) = \eta$ on $[\beta_k, 1]$. Since $\mathrm{ESS}_k$ is non-increasing in $\beta$, this root can be found by bisection. We take the left end of the final bisection bracket, where $\mathrm{ESS}_k \geq \eta$, so each step is slightly conservative. The number of temperatures $K$ is therefore determined by $\eta$, $N_{\mathrm{SMC}}$ and the target, not set in advance.

The population is then resampled by systematic resampling, applied after a uniformly random permutation of the particles, and moved by MCMC targeting the newly selected distribution $\pi_{\beta_{k+1}}$. This is the standard adaptive resample-move construction (Del Moral et al., 2006; Jasra et al., 2011; Gilks & Berzuini, 2001).

We use preconditioned one-step HMC with full momentum refresh as our MCMC kernel. Let $\widehat{\Sigma}_k$ be the symmetrized empirical covariance of the resampled particles at $\beta_k$ and $L_k$ its Cholesky factor. If $\widehat{\Sigma}_k$ is not positive definite, we use $L_k = \mathrm{diag}(\widehat{\Sigma}_k)^{1/2}$ instead. Given a step size $\varepsilon_k$, one move from $x$ is

$$\begin{aligned} v &\sim \mathcal{N}(0, I_d), & x_{1/2} &= x + \frac{\varepsilon_k}{2} L_k v, \\ v' &= v + \varepsilon_k L_k^\top \nabla \log \pi_{\beta_k}(x_{1/2}), & x' &= x_{1/2} + \frac{\varepsilon_k}{2} L_k v'. \end{aligned} \tag{A.2}$$

where $\nabla \log \pi_{\beta_k} = \nabla \log \rho - \beta_k \nabla E$. The move is accepted with probability

$$\alpha(x, x') = \min\left\{1, \exp\left(\log \pi_{\beta_k}(x') - \tfrac{1}{2}\|v'\|^2 - \log \pi_{\beta_k}(x) + \tfrac{1}{2}\|v\|^2\right)\right\}. \tag{A.3}$$

For fixed $L_k$, this is ordinary HMC after the linear change of coordinates $x \mapsto L_k^{-1}x$; the Metropolis acceptance rule preserves $\pi_{\beta_k}$. The step size $\varepsilon_k$ is chosen by bisection so that the mean acceptance probability of one trial move over the population lies in $[0.40, 0.65]$. Each particle then takes $n_{\mathrm{MCMC}}$ moves (specified in Appendix C.5). The adapted parameters $(\varepsilon_k, L_k)$ are then frozen and reused for rejuvenation at $\beta_k$ during self-training. Each subsequent kernel therefore leaves its target exactly invariant, although invariance alone does not guarantee rapid inter-basin mixing.

The populations obtained at $\beta_1, \dots, \beta_K$ approximate the corresponding targets and initialize $\mathcal{D}_1, \dots, \mathcal{D}_K$ with equal weights; $\mathcal{D}_0$ is filled with independent draws from $\rho$. For DW-4 and LJ-13, particles are represented in ambient coordinates and the retained populations are centered before entering the replay buffers. Appendix C.3 explains the centered target for each system and the corresponding sampling convention.

### A.2 Replay buffers and per-batch weight normalization

Each buffer has capacity $C = N_{\mathrm{SMC}}$. $\mathcal{D}_0$ contains independent draws from $\rho$; $\mathcal{D}_1, \dots, \mathcal{D}_K$ are initialized with the retained SMC populations. During training, $\mathcal{D}_0$ and $\mathcal{D}_1$ remain fixed, other buffers are updated every iteration. Specifically, at each active temperature $\beta_k$, $k = 1, \dots, K-1$, we generate $N$ new proposal samples per refresh, correct them with importance resampling, rejuvenate with HMC, and insert the resulting weighted samples into $\mathcal{D}_{k+1}$. Thus, only $\mathcal{D}_2, \dots, \mathcal{D}_K$ have their initial samples removed through first-in, first-out (FIFO) replacement. They are fully replaced after $\lceil C/N \rceil$ refreshes, subject to the convention for removing partially retained batches.

After resampling according to the proposal importance weights and applying MCMC at $\beta_k$, let $\ell_i = -(\beta_{k+1} - \beta_k)E(x_i)$ be the bridge log weights. Store

$$\widetilde{\ell}_i = \ell_i - \mathrm{logsumexp}(\ell_1, \dots, \ell_N) + \log N. \tag{A.4}$$

Here $\mathrm{logsumexp}(\ell_1, \dots, \ell_N) = \log \sum_i e^{\ell_i}$. Each batch has total weight $N$, matching an equally sized batch of initial samples with unit weights, while its relative weights are unchanged. Thus each refresh contributes a self-normalized approximation of the target. Without this normalization, fluctuations in the batch estimate of $Z_{\beta_{k+1}}/Z_{\beta_k}$ would also determine its influence relative to other batches.

We use systematic resampling both to correct proposals and to draw training minibatches. The resulting samples can be dependent. Per-batch normalization and additional optimization do not make the finite adaptive buffer an exact sample from its target, and we do not claim convergence of the self-training loop at fixed buffer size.

At each optimization step, we draw a minibatch of $b$ samples from each buffer $\mathcal{D}_k$ by systematic resampling according to the stored weights, sample fresh diffusion times $t \sim \mathrm{Unif}[t_{\min}, T]$ and perturbations $\varepsilon \sim \mathcal{N}(0, I_d)$, and take one Adam step on (5). We perform $J$ such steps between consecutive buffer refreshes. Energy gradients are computed once per stored sample and cached; no gradients pass through the procedure that generates or selects the samples. The value of $J$, the number of outer iterations and the optimizer settings are listed in Appendix C.5.

### A.3 Numerical generation and stable log weights

The path weight can be accumulated during sampling without storing the trajectory. Write $h = T/L$, $a = e^{-h}$, $\sigma = \sqrt{1 - a^2}$, and $c_n = 2(1 - a)r_\theta(y_n, T - nh, \beta)$. One step is

$$y_{n+1} = ay_n + c_n + \sigma\varepsilon_n, \qquad \varepsilon_n \sim \mathcal{N}(0, I_d).$$

The proposal and reference transitions have the same covariance, so their Gaussian normalizers cancel. The log-weight increment is simply

$$\Delta_n = \log \frac{g_h(y_{n+1} \mid y_n)}{q_{\theta,n}(y_{n+1} \mid y_n, \beta)} = \frac{\|y_{n+1} - ay_n - c_n\|^2 - \|y_{n+1} - ay_n\|^2}{2\sigma^2} = -\frac{c_n^\top \varepsilon_n}{\sigma} - \frac{\|c_n\|^2}{2\sigma^2}.$$

The last expression uses the noise already drawn for the step and avoids subtracting two squared norms. Starting from zero, accumulate these increments and add the terminal energy contribution:

$$\log w = \sum_{n=0}^{L-1} \Delta_n - \beta E(y_L). \tag{A.5}$$

This is algebraically the same weight as (7). When $r_\theta \equiv 0$, every increment is zero; at $\beta = 0$, the full log weight is zero for any step size.

**Algorithm 2** Gaussian-reference sampler with online log weights

**Input:** $r_\theta$, energy $E$, temperature $\beta$, horizon $T$, steps $L$

$h \leftarrow T/L$, $a \leftarrow e^{-h}$, $\sigma \leftarrow \sqrt{1 - e^{-2h}}$
$y \sim \mathcal{N}(0, I_d)$, $\ell \leftarrow 0$
**for** $n = 0, \dots, L-1$ **do**
  $c \leftarrow 2(1-a)\, r_\theta(y, T - nh, \beta)$; $\varepsilon \sim \mathcal{N}(0, I_d)$
  $y' \leftarrow ay + c + \sigma\varepsilon$
  $\ell \leftarrow \ell - c^\top \varepsilon/\sigma - \|c\|^2/(2\sigma^2)$
  $y \leftarrow y'$
$\ell \leftarrow \ell - \beta E(y)$
**return** $(y, \ell)$

Weights are normalized in log space. At positive $\beta$, the terminal energy term downweights high-energy endpoints; an energy evaluated as $+\infty$ gives zero weight when the transition terms are finite. Resampling requires at least one finite log weight, and a batch with none must be treated as a failed refresh rather than normalized. This numerical issue is separate from the validity of the exact weight formula.

We use the diffusion horizon $T = 4$. On a grid of step size $h$, the smallest time queried by the network is $t = h$; it is never evaluated at $t = 0$. The finer evaluation grid places a few final queries below the training cutoff $t_{\min}$. This extrapolation can affect raw-sample accuracy and weight variance, but not the validity of the importance weights for the numerical proposal actually used. The training grid does not query the network below $t_{\min}$.

## B JUSTIFICATION OF THE SCORE FORMULA

**The regression target.** Fix $\beta$ and let $k_t(z \mid x) = \mathcal{N}(z; \alpha_t x, (1 - \alpha_t^2) I_d)$, with $\alpha_t = e^{-t}$. Assume that $E$ is differentiable and that differentiation under the integrals and the integration by parts below are valid. The Gaussian kernel satisfies $\rho(x) k_t(z \mid x) = \rho(z) k_t(x \mid z)$. Consequently, for $X \sim \pi_\beta$ and $X_t \mid X \sim k_t(\cdot \mid X)$,

$$\frac{\pi_{\beta,t}(z)}{\rho(z)} = \frac{1}{Z_\beta} \int e^{-\beta E(x)} k_t(x \mid z)\, \mathrm{d}x.$$

Since $\nabla_z k_t(x \mid z) = -\alpha_t \nabla_x k_t(x \mid z)$, integration by parts gives

$$\nabla_z \log \frac{\pi_{\beta,t}(z)}{\rho(z)} = \alpha_t \frac{\int \nabla_x e^{-\beta E(x)} k_t(x \mid z)\, \mathrm{d}x}{\int e^{-\beta E(x)} k_t(x \mid z)\, \mathrm{d}x} = -\alpha_t \beta\, \mathbb{E}[\nabla E(X) \mid X_t = z].$$

The final equality follows because the conditional density of $X$ given $X_t = z$ is proportional to $e^{-\beta E(x)} k_t(x \mid z)$. This is the relative-score identity used in (4) (Bortoli et al., 2024; Phillips et al., 2024). Least-squares regression learns the conditional mean of its target. Thus, when $\mathbb{E}_{\pi_\beta} \|\nabla E(X)\|^2 < \infty$ for $\beta > 0$, the population minimizer with regression target $-\alpha_t \beta \nabla E(X)$ is $r^*(X_t, t, \beta)$. At $\beta = 0$, both are zero. Finite-buffer sampling and the target clipping used in the particle experiments are approximations to this population objective, not part of the identity.

**The reverse sampler.** The forward Ornstein–Uhlenbeck process has drift $-x$ and noise coefficient $\sqrt{2}$. Its standard reverse-time formula (Song et al., 2021), written with sampling time $s$ increasing from 0 to $T$, has drift

$$x + 2\nabla_x \log \pi_{\beta, T-s}(x) = -x + 2r^*(x, T - s, \beta).$$

With the exact score and initial distribution $\pi_{\beta,T}$, this process ends at $\pi_\beta$. The explicit perturbation $X_T = \alpha_T X + \sqrt{1 - \alpha_T^2}\, \varepsilon$ approaches the Gaussian reference as $T$ increases, motivating initialization from $\rho$. At finite $T$, this replacement is an approximation. The discrete importance weights in (7) remain valid for the numerical sampler regardless of this initial mismatch or score error.

# C EXPERIMENTAL DETAILS

## C.1 TARGET DEFINITIONS AND REFERENCE DISTRIBUTIONS

**Gaussian mixture model (GMM).** We consider a synthetic target on $\mathbb{R}^d$ with independent coordinates, each following a two-component Gaussian mixture:

$$\pi(x) = \prod_{j=1}^{d} \left[ w_1 \mathcal{N}(x_j; \mu_1, \sigma^2) + w_2 \mathcal{N}(x_j; \mu_2, \sigma^2) \right],$$

where $(\mu_1, \mu_2) = (-1.5, 2.5)$, $(w_1, w_2) = (0.2, 0.8)$, and $\sigma = 0.5$. We use $d = 16$ and $d = 50$, keeping the per-coordinate parameters fixed so that the two settings differ only in dimension. This construction yields $2^d$ well-separated modes with unequal probability masses. The main challenge is to explore these modes and recover their relative masses despite the low-density regions separating them. Exact independent samples can be obtained by sampling the mixture component and Gaussian noise independently for each coordinate.

**4-particle double-well (DW-4).** DW-4 is an interacting-particle benchmark for sampling multimodal Boltzmann distributions (Köhler et al., 2020). A configuration consists of four particles $x^1, \dots, x^4 \in \mathbb{R}^2$, represented by $x \in \mathbb{R}^8$. The target density is

$$\pi(x) \propto \exp[-\tau U_{\mathrm{DW}}(x)], \quad U_{\mathrm{DW}}(x) = \sum_{1 \le i < j \le 4} \left[ a(r_{ij} - r_0)^4 + b(r_{ij} - r_0)^2 + c \right],$$

where $r_{ij} = \|x^i - x^j\|_2$ and $\tau$ is the physical inverse temperature. We set $a = 0.9$, $b = -4$, $c = 0$, $r_0 = 4$, and $\tau = 1$. Since the energy is translation-invariant, we impose $\sum_{i=1}^{4} x^i = 0$ and define the density with respect to Lebesgue measure on this six-dimensional subspace. The two preferred pairwise distances give rise to competing particle arrangements and multiple energy minima. The main sampling challenge is to cross the energy barriers separating these arrangements. As reference samples, we use the MCMC samples of Klein et al. (2023), obtained from the DEM repository of Akhound-Sadegh et al. (2024).

**13-particle Lennard–Jones (LJ-13).** LJ-13 models a cluster of thirteen particles $x^1, \dots, x^{13} \in \mathbb{R}^3$, with configurations represented by $x \in \mathbb{R}^{39}$. The Lennard–Jones potential models short-range repulsion and longer-range attraction between particles and provides a standard benchmark for sampling atomic cluster configurations (Köhler et al., 2020). We use the confined, regularized target

$$\pi(x) \propto \exp[-\tau U_{\mathrm{LJ}}(x)], \quad U_{\mathrm{LJ}}(x) = \varepsilon \sum_{1 \le i < j \le 13} \left[ \left( \frac{r_m}{\widetilde{r}_{ij}} \right)^{12} - 2 \left( \frac{r_m}{\widetilde{r}_{ij}} \right)^{6} \right] + \frac{\kappa}{2} \sum_{i=1}^{13} \|x^i\|_2^2,$$

where $\widetilde{r}_{ij} = \sqrt{\|x^i - x^j\|_2^2 + \delta}$. We set $\varepsilon = 2$, $r_m = 1$, $\kappa = 1$, $\delta = 10^{-5}$, and the physical inverse temperature $\tau = 1$. The harmonic term confines the cluster, while $\delta$ regularizes the repulsive interaction near particle collisions. We impose $\sum_{i=1}^{13} x^i = 0$ and define the density with respect to Lebesgue measure on the resulting 36-dimensional subspace. The main sampling difficulties are transitions between metastable cluster configurations and the steep energy gradients associated with short-range repulsion. As for DW-4, the reference samples are the MCMC samples of Klein et al. (2023), obtained from the DEM repository of Akhound-Sadegh et al. (2024).

For the particle targets and the Gaussian reference, the relative energy used by the sampler is $E(x) = \tau U(x) - \|x\|^2/2$, up to an additive constant, with $\tau = 1$ in these experiments.

## C.2 NEURAL NETWORK ARCHITECTURE

All experiments use a single network $r_\theta(x_t, t, \beta)$ shared across the whole temperature ladder, conditioned on both the diffusion time $t$ and the temperature parameter $\beta$. The architecture is chosen to match the structure of each target. We use a residual MLP for the Gaussian mixtures and an equivariant graph network for the particle systems. On centered particle configurations, the target is unchanged by permutations of identical particles, rotations, and reflections.

**Gaussian mixtures.** The relative score network $r_\theta(x_t, t, \beta)$ is a residual MLP with input $[x_t,\ \mathrm{emb}_{16}(t),\ \mathrm{emb}_8(\beta)]$, where $\mathrm{emb}_m$ denotes an $m$-dimensional sinusoidal embedding with log-spaced frequencies in $[1, 10]$ for $t$ and $[0.5, 5]$ for $\beta$. A linear layer maps the input to width 256, followed by three pre-norm residual blocks (RMSNorm, Dense, ELU, Dense) and a linear output layer. The output weights are initialized from $\mathcal{N}(0, \sigma^2)$ with $\sigma^2 = 10^{-8}$, and the output bias is set to zero, so that $r_\theta \approx 0$ at initialization, consistent with the zero relative score at $\beta = 0$ (Section 3.1). The network has $4.1 \times 10^5$ parameters for $d = 16$ and $4.3 \times 10^5$ for $d = 50$.

**Particle systems.** For DW-4 and LJ-13, $r_\theta$ is an E($n$)-equivariant graph neural network (EGNN) (Satorras et al., 2021) on the fully connected particle graph, with 5 layers of width 128, SiLU activations, attention-gated messages, and residual node updates. Each node receives $t$ as a scalar feature together with an 8-dimensional sinusoidal embedding of $\beta$ (frequencies in $[0.5, 5]$), and each edge carries the squared inter-particle distance of the input configuration. Coordinate updates are bounded by $\tanh$, with a per-layer range of 3. The output is the total coordinate displacement with its mean removed. It is therefore equivariant to permutations, rotations, and reflections, invariant to translations, and lies in the zero center-of-mass subspace (Appendix C.3). The network has $5.8 \times 10^5$ parameters.

### C.3 Projection onto the zero center-of-mass subspace

For DW-4 and LJ-13, we represent a configuration of $k$ particles in $\mathbb{R}^s$ as $y = (y^1, \dots, y^k) \in \mathbb{R}^{ks}$. We remove its center of mass by subtracting the mean particle position:

$$x^i = y^i - \frac{1}{k}\sum_{j=1}^{k} y^j, \qquad i = 1, \dots, k. \tag{C.1}$$

Equivalently, $x = \mathcal{A}y$, where

$$\mathcal{A} = \left(I_k - \frac{1}{k}\mathbf{1}_k\mathbf{1}_k^\top\right) \otimes I_s \tag{C.2}$$

is the orthogonal projection onto $\mathcal{X}^{\mathrm{CoM}} = \{x \in \mathbb{R}^{ks} : \sum_i x^i = 0\}$. This subspace has dimension $(k-1)s$. All diffusion sampling and density calculations below are understood on this space. The Gaussian reference is standard in an orthonormal coordinate system on the subspace; in ambient coordinates its covariance is $\mathcal{A}$.

For DW-4, the physical energy depends only on pairwise distances. Fixing the center of mass removes the unconstrained translation and makes the target a distribution over internal configurations. For LJ-13, the pair interaction is also translation-invariant, but the full potential is not, because it includes harmonic confinement. The confinement separates as

$$\sum_{i=1}^{k} \|y^i\|^2 = \sum_{i=1}^{k} \|y^i - \bar{y}\|^2 + k\|\bar{y}\|^2, \qquad \bar{y} = \frac{1}{k}\sum_{i=1}^{k} y^i.$$

Thus the LJ-13 density factors into a center-of-mass term and an internal-coordinate term. Centering a draw preserves its internal-coordinate distribution; it does not assert translation invariance of the full confined target.

**HMC steps.** We use preconditioned HMC during initialization and training. Retained SMC particles are centered before entering the replay buffers; during training, the same centering is applied after each HMC move. The target-invariance claim requires the proposal and acceptance calculation to be consistent with the centered configuration space; projection alone does not establish it.

**Diffusion steps.** We center the Gaussian noise used in both training and sampling. For a centered buffer sample $x_0$, the perturbation is

$$x_t = \alpha_t x_0 + \sqrt{1 - \alpha_t^2}\,\mathcal{A}\varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, I_{ks}).$$

The score network also returns a centered vector, so sampling takes the form

$$y_{n+1} = a y_n + 2(1-a) r_\theta(y_n, T - nh, \beta) + \sqrt{1 - a^2}\,\mathcal{A}\varepsilon_n.$$

Starting from a centered reference draw, all states remain centered. The learned and reference transitions have the same covariance $(1-a^2)\mathcal{A}$ in ambient coordinates, or $(1-a^2)I$ in coordinates on the subspace. Their Gaussian normalizers therefore cancel, and the log-weight formula in Appendix A.3 applies with the projected noise.

### C.4 Reported metrics

We assess sample quality using three complementary discrepancies: Distance-$\mathcal{W}_2$, Energy-$\mathcal{W}_2$, and Geometric-$\mathcal{W}_2$, as considered in the evaluation of PITA (Akhound-Sadegh et al., 2025). We specify our definitions and how we implemented these metrics below. Lower values indicate better agreement with the reference samples.

**Distance-$\mathcal{W}_2$.** Let $h(x) \in \mathbb{R}^f$ denote a vector of summary statistics. For the Gaussian mixture targets, $h(x)$ consists of the coordinates of $x$; for DW-4 and LJ-13, it consists of all pairwise interparticle distances. Given generated samples $\{x_i\}_{i=1}^n$ and reference samples $\{y_j\}_{j=1}^m$, we pool these statistics into the one-dimensional empirical distributions

$$\widehat{\mu}_h = \frac{1}{nf}\sum_{i=1}^{n}\sum_{\ell=1}^{f}\delta_{h_\ell(x_i)}, \qquad \widehat{\nu}_h = \frac{1}{mf}\sum_{j=1}^{m}\sum_{\ell=1}^{f}\delta_{h_\ell(y_j)},$$

and report

$$\text{Distance-}\mathcal{W}_2 = \mathcal{W}_2(\widehat{\mu}_h, \widehat{\nu}_h).$$

The one-dimensional 2-Wasserstein distance admits the quantile representation (Peyré & Cuturi, 2018)

$$\mathcal{W}_2(\mu,\nu) = \left(\int_0^1 \left|F_\mu^{-1}(u) - F_\nu^{-1}(u)\right|^2 \mathrm{d}u\right)^{1/2},$$

where $F_\mu^{-1}$ and $F_\nu^{-1}$ are the corresponding quantile functions. Distance-$\mathcal{W}_2$ measures agreement in the pooled coordinate or distance distribution, but does not capture dependencies among statistics within each configuration.

**Energy-$\mathcal{W}_2$.** Following the energy-distribution evaluation used by Akhound-Sadegh et al. (2025), we compare the empirical distributions of the energy $E(x)$:

$$\text{Energy-}\mathcal{W}_2 = \mathcal{W}_2\left(\frac{1}{n}\sum_{i=1}^{n}\delta_{E(x_i)}, \frac{1}{m}\sum_{j=1}^{m}\delta_{E(y_j)}\right).$$

This metric measures whether generated samples reproduce the reference energy distribution, complementing Distance-$\mathcal{W}_2$. For both one-dimensional metrics, equal-sized scalar sample sets are compared exactly by sorting and computing the root mean squared difference. For unequal sizes, we approximate the quantile integral using 1000 uniformly spaced midpoint quantiles. The current evaluation excludes non-finite scalar values before either calculation. These metrics therefore do not penalize numerical failures: when exclusions occur, they describe only the retained finite subset and can understate the error of the full sample set.

**Geometric-$\mathcal{W}_2$.** Distance-$\mathcal{W}_2$ compares pooled statistics and so cannot tell whether individual configurations are correct. For DW-4 and LJ-13, we therefore also compare their whole configurations. Two configurations that differ only by relabelling identical particles, or by a rotation or reflection, describe the same physical state, so we measure the distance between them after aligning out these symmetries. For configurations $X, Y \in \mathbb{R}^{k\times s}$ ($k$ particles in $s$ dimensions, centred at the origin), the squared distance is

$$d^2(X,Y) = \min_{P,\,R} \|X - PYR\|_{\mathrm{F}}^2,$$

where $P$ ranges over particle permutations and $R$ over rotations and reflections. Geometric-$\mathcal{W}_2$ is the 2-Wasserstein distance between the generated and reference samples with $d$ as the ground distance.

We approximate the joint alignment: following Köhler et al. (2020), we match particles by a linear assignment and then fit a rotation or reflection to the matched coordinates by least squares (orthogonal

Procrustes). The resulting squared distance is an upper bound on $d^2(X, Y)$. We use these approximate alignment costs for the full sets of 10,000 generated and 10,000 reference samples. The outer transport problem is solved exactly for this cost matrix; the configuration alignment remains approximate.

### C.5 Numerical settings and baselines reimplementation

We list here the settings needed to reproduce our results. The SMC particle count $N_{\mathrm{SMC}}$ also sets the replay-buffer capacity $C$, and $K$ denotes the number of temperatures chosen by the adaptive sweep (Appendix A.1). The fixed training lengths below were selected using validation performance, as were the baseline training lengths. We report the cost of these final checkpoints, rather than time to a shared error tolerance or a formal convergence threshold.

**GMM16 ($d = 16$).** The SMC sweep uses $N_{\mathrm{SMC}} = 81{,}920$ particles, ESS threshold $\eta = 0.5$, and 3 HMC steps per temperature, giving $K = 12$ for all three seeds. Training runs for 10 outer iterations. Each iteration generates $N = 30{,}000$ proposals per temperature with $L_{\mathrm{train}} = 100$ sampler steps, applies $n_{\mathrm{MCMC}} = 10$ HMC steps after resampling, and then takes $J = 10^3$ Adam steps. For final evaluation, we generate $10^5$ fresh samples with $L_{\mathrm{eval}} = 2000$ steps and compare them with $10^5$ exact draws from the target.

**GMM50 ($d = 50$).** The per-coordinate target is the same as in GMM16, but the higher dimension calls for more MCMC effort at initialization and longer training. The SMC sweep uses $N_{\mathrm{SMC}} = 81{,}920$, $\eta = 0.1$, and 50 HMC steps per temperature, giving $K = 11$ for all three seeds. Training runs for 10 outer iterations with $N = 30{,}000$, $L_{\mathrm{train}} = 100$, $n_{\mathrm{MCMC}} = 20$, and $J = 10^4$. Evaluation follows GMM16, with $10^5$ fresh samples, $L_{\mathrm{eval}} = 2000$, and $10^5$ exact reference draws.

**DW-4 ($d = 8$).** The SMC sweep uses $N_{\mathrm{SMC}} = 81{,}920$, $\eta = 0.2$, and 10 HMC steps per temperature, giving $K = 6$ for all three seeds. Training runs for 10 outer iterations with $N = 30{,}000$, $L_{\mathrm{train}} = 100$, $n_{\mathrm{MCMC}} = 10$, and $J = 10^3$. To keep training stable under large energy gradients, we cap the norm of the regression target at 20 and clamp the network output to $[-10^4, 10^4]$. For final evaluation, we generate $10^4$ fresh samples with $L_{\mathrm{eval}} = 2000$ steps.

**LJ-13 ($d = 39$).** The SMC sweep uses $N_{\mathrm{SMC}} = 25{,}000$, $\eta = 0.2$, and 200 HMC steps per temperature, giving $K = 7$ for all three seeds. Training runs for 50 outer iterations with $N = 5{,}000$, $L_{\mathrm{train}} = 25$, $n_{\mathrm{MCMC}} = 2000$, and $J = 10^3$. We cap the norm of the regression target at 150 per particle and clamp the network output to $[-10^4, 10^4]$. For final evaluation, we generate $10^4$ fresh samples with $L_{\mathrm{eval}} = 1000$ steps.

**Baselines reimplementation.** We reimplement both methods in JAX based on the authors' released code, and check every numerical component against reference outputs recorded from the original PyTorch implementation. Where available, we use settings from the authors' released code; the benchmark-specific choices are described below. Since the original code provides no configuration for GMM or DW-4, we choose the temperature ladders for these targets ourselves, using the same maximum temperature $\mathsf{T}_{\max}$ for both methods. For GMM we take $\mathsf{T}_{\max} = 30$ from TA-BG's GMM experiment. For DW-4 we set $\mathsf{T}_{\max} = 10$ to reduce the barriers faced during initial reverse-KL training. PITA uses 6 geometrically spaced temperatures from $\mathsf{T}_{\max}$ down to $\mathsf{T} = 1$, matching the number of temperatures in its original LJ-13 ladder. For these targets, PITA's initial samples at $\mathsf{T}_{\max}$ are generated by MALA rather than taken from molecular-dynamics data. TA-BG uses its original GMM ladder of 7 geometric steps from 30 to 1 on GMM, and 9 geometric steps from 10 to 1 on DW-4, following its alanine-dipeptide recipe. On DW-4, TA-BG's flow acts on Cartesian coordinates with the centre of mass removed. For each method, we report the final checkpoint selected using validation performance. On LJ-13, both baselines diverged on our hardware, so we report the results published by Akhound-Sadegh et al. (2025).