Flow Perturbation++: Multi-Step Unbiased Jacobian Estimation for High-Dimensional Boltzmann Sampling

Xin Peng[1,2] **Ang Gao**[1*]
[1]State Key Laboratory of Information Photonics and Optical Communications,
School of Physical Science and Technology,
Beijing University of Posts and Telecommunications (BUPT)
[2]Shanghai Academy of AI for Science (SAIS)
xinpeng@bupt.edu.cn anggao@bupt.edu.cn

Abstract

The scalability of continuous normalizing flows (CNFs) for unbiased Boltzmann sampling remains limited in high-dimensional systems due to the cost of Jacobian-determinant evaluation, which requires D backpropagation passes through the flow layers. Existing stochastic Jacobian estimators such as the Hutchinson trace estimator reduce computation but introduce bias, while the recently proposed Flow Perturbation method is unbiased yet suffers from high variance. We present **Flow Perturbation++**, a variance-reduced extension of Flow Perturbation that discretizes the probability-flow ODE and performs unbiased stepwise Jacobian estimation at each integration step. This multi-step construction retains the unbiasedness of Flow Perturbation while achieves substantially lower estimator variance. Integrated into a Sequential Monte Carlo framework, Flow Perturbation++ achieves significantly improved equilibrium sampling on a 1000D Gaussian Mixture Model and the all-atom Chignolin protein compared with Hutchinson-based and single-step Flow Perturbation baselines.

1 Introduction

Sampling from the Boltzmann distribution $p(\mathbf{x}) \propto e^{-u(\mathbf{x})}$ is a long-standing challenge in statistical mechanics and molecular modeling. Rugged, high-dimensional energy landscapes with numerous metastable states are separated by large free-energy barriers [14], causing classical Molecular Dynamics (MD) [57] and Markov Chain Monte Carlo (MCMC) [39] methods to mix extremely slowly. Enhanced sampling techniques—such as replica exchange [20, 52], umbrella sampling [55], metadynamics [28], and transition-path sampling [9]—can mitigate these difficulties but often introduce substantial computational overhead or require carefully engineered, system-specific collective variables.

Beyond physical systems, Boltzmann distributions appear as a common functional form in statistical machine learning and reinforcement learning (RL) [38, 16, 6]. In RL, such forms arise in entropy-regularized and inference-based formulations, e.g., maximum-entropy RL induces policies $\pi(a \mid s) \propto \exp\big(Q^\pi(s,a)/\alpha\big)$ [15, 48, 22], and trajectory-level methods define $\pi(\mathbf{x}) \propto \exp\big(\beta\, R(\mathbf{x})\big)$ [5, 35, 62]. However, in these settings this form is mainly descriptive rather than used for explicit Boltzmann sampling. The prevalence of Boltzmann distributions motivates the development of efficient methods for sampling from these distributions.

This widespread demand has driven the development of deep generative models for learning Boltzmann target distributions and enabling efficient sampling. By transporting a simple base distribution

*Corresponding author

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

to the target, such models have achieved notable success in molecular and material design, including drug-like molecule generation, protein conformations, and novel materials [61, 1, 60, 19, 59, 41, 54].

Normalizing flows (NFs) [45, 11, 12, 23, 54] provide exact likelihood evaluation through invertible transformations between a simple prior and a complex target distribution. Discrete NFs rely on carefully designed architectures to make Jacobian determinants tractable, trading expressive flexibility for computational efficiency.

Continuous normalizing flows (CNFs) [26], a continuous-time limit of NFs, parameterize transformations via neural ordinary differential equations (neural ODEs) [7] and have recently been applied to Boltzmann sampling [41, 25, 58, 10]. A key advantage of CNFs is their ease of training, enabled by simulation-free objectives such as diffusion-based probability-flow ODEs [18, 50] and flow matching [31, 33, 2], which have achieved strong empirical performance in image generation and molecular modeling.

Despite these advantages, unbiased Boltzmann sampling with CNFs remains challenging. To obtain unbiased expectations from the Boltzmann target, one must correct the flow-induced proposal distribution via importance reweighting, $w(\mathbf{x}) = p(\mathbf{x})/q(\mathbf{x})$, which requires evaluating likelihood ratios. Evaluating importance weights requires integrating the trace of the Jacobian of the velocity field along the flow trajectory, necessitating D backpropagation passes through all layers of the flow, where D is the dimensionality of the system [41]. While stochastic trace estimators such as the Hutchinson estimator [21] reduce this cost [58], exponentiating their stochastic trace estimates can introduce convergence issues and systematic bias [41, 13]. Consequently, existing CNF-based methods have only achieved unbiased Boltzmann sampling for relatively low-dimensional systems [25].

Building on Flow Perturbation (FP) [43], which estimates flow Jacobians via stochastic perturbations to reduce the cost of unbiased Boltzmann sampling, we introduce **Flow Perturbation++ (FP++)**, a theoretically grounded, determinant-unbiased estimator for continuous-flow dynamics under exact inverse-step derivatives, with variance reduction under the conditions in Appendix C. FP++ exploits the semigroup property of flow maps to decompose the global Jacobian $|\det J_{1\to T}| = \prod_{k=1}^{T} |\det J_k|$, estimating each factor without computing the full Jacobian directly.

Our main contributions are:

- **Unbiased determinant estimation.** FP++ retains the determinant-level unbiasedness of FP [43] for exact inverse-step maps and directional derivatives, while improving numerical stability.
- **Variance reduction.** Under the conditions in Appendix C, decomposing the global Jacobian into per-step determinants reduces log-domain variance by removing probe-induced correlations between steps, especially for flows with many integration steps.
- **Seamless SMC integration.** FP++ naturally integrates with Sequential Monte Carlo (SMC) annealing [32], supporting stable Boltzmann reweighting and mitigating particle degeneracy.
- **Scalable validation.** Experiments on a 1000-dimensional Gaussian mixture and an all-atom Chignolin mutant show FP++ improves equilibrium sampling over Hutchinson and original FP baselines.

2 Related work

2.1 Balancing expressivity and tractability in flows

Generative models for Boltzmann sampling differ primarily in how they handle the Jacobian determinant.

Constrained architectures. Early flow models such as NICE [11], RealNVP [12], and Glow [23] use structured transformations (e.g., affine coupling) that yield triangular Jacobians and allow $\mathcal{O}(D)$ determinant evaluation. While computationally efficient, these architectural constraints limit expressivity and are often insufficient for complex molecular or high-dimensional target distributions [41].

Free-form continuous flows. More flexible approaches—including Neural ODEs [7], diffusion-based Probability Flows [18, 50], flow-matching methods [31, 33, 2], and i-ResNets [4]—remove

structural constraints to model richer dynamics. i-ResNets enforce invertibility via residual blocks and estimate determinant product with Hutchinson's trace estimator, while diffusion and flow-matching flows learn vector fields without explicit Jacobian computations during training. However, exact likelihood and unbiased inference still require evaluating Jacobian determinants, leaving a fundamental trade-off between expressivity and tractability.

Complementary sampler pipelines. FALCON learns few-step invertible maps [44], whereas FAB and SBG-SMC improve learned proposals, annealing, or sequential weighting [40, 53]. These methods address proposal construction and particle transport, while their likelihood or importance weights still depend on a Jacobian correction. They therefore form a parallel and complementary line to Jacobian and likelihood estimators such as Hutchinson and FP [21, 43]. FP++ belongs to the latter line, which is the focus of our direct comparisons.

2.2 Sequential Monte Carlo

High-dimensional importance sampling often suffers from weight collapse. Sequential Monte Carlo (SMC) [32] mitigates this via tempering and resampling. Flow-SMC approaches [3, 51] combine learned flows with SMC to stabilize weight evolution. FP++ integrates naturally with SMC annealing, providing a determinant-based Boltzmann reweighting correction in high dimensions.

3 Background

3.1 Boltzmann reweighting

Given a generative model $q(\mathbf{z})$ induced by a mapping $\mathbf{f} : \mathbf{z} \mapsto \mathbf{x}$ ($\mathbf{z} \sim q(\mathbf{z})$), the unbiased Boltzmann expectation of an observable $O(\mathbf{x})$ is

$$\mathbb{E}_{\mathbf{x}\sim p}[O] = \frac{\mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})}\left[O(\mathbf{f}(\mathbf{z}))\, e^{-W(\mathbf{z})}\right]}{\mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})}\left[e^{-W(\mathbf{z})}\right]}, \tag{1}$$

where the generalized work is $W(\mathbf{z}) = u_X(\mathbf{f}(\mathbf{z})) - u_Z(\mathbf{z}) - \Delta S(\mathbf{z})$, with $u_X(\mathbf{x})$ and $u_Z(\mathbf{z})$ denoting the dimensionless potentials of the target Boltzmann and prior distributions, respectively. The entropic contribution $\Delta S(\mathbf{z}) = \log\left|\det\left(\frac{\partial \mathbf{f}}{\partial \mathbf{z}}\right)\right|$ is determined by the Jacobian of the generative map and represents the primary computational bottleneck. In practice, the importance weights $e^{-W(\mathbf{z})}$ are normalized.

3.2 Probability flow ODEs

Diffusion-based generative models define an SDE $d\mathbf{x}_t = b(\mathbf{x}_t, t)\, dt + g(t)\, d\mathbf{w}_t$. The corresponding probability flow ODE [50] is

$$\frac{d\mathbf{x}_t}{dt} = b(\mathbf{x}_t, t) - \frac{1}{2} g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x}_t), \tag{2}$$

which defines a special type of continuous normalizing flow (CNF). During training, score-matching allows the model to avoid direct computation of the Jacobian determinant, which would be prohibitively expensive in high dimensions.

At inference time, evaluating importance weights or likelihoods requires the log-determinant of the flow Jacobian [26]:

$$\log\left|\det\left(\frac{\partial \mathbf{f}}{\partial \mathbf{z}}\right)\right| = \int_0^T \nabla_{\mathbf{x}} \cdot v(\mathbf{x}_t, t)\, dt, \tag{3}$$

where $v(\mathbf{x}_t, t) = b(\mathbf{x}_t, t) - \frac{1}{2} g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x}_t)$. Here, $\nabla_{\mathbf{x}} \cdot v$ denotes the divergence of the velocity field, which can be obtained as the trace of its Jacobian. This term is computationally expensive in high dimensions, which motivates estimating the divergence using the Hutchinson estimator (see Appendix E).

3.3 Jacobian determinant estimation

Continuous flows require Jacobian determinant for likelihood evaluation or importance sampling. Hutchinson's estimator [21] approximates traces via randomized projections,

$$\hat{T}_{\text{Hutch}} = \mathbf{u}^\top \left(\frac{\partial v}{\partial \mathbf{x}}\right) \mathbf{u}, \quad \mathbf{u} \sim \mathcal{D},\ \mathbb{E}[\mathbf{u}\mathbf{u}^\top] = \mathbf{I}, \tag{4}$$

Here $\mathcal{D}$ is a probe distribution on $\mathbb{R}^D$, such as the standard Gaussian distribution or independent Rademacher entries; $\mathbf{I}$ is the identity matrix. The trace estimate is unbiased, but exponentiation generally biases determinant estimates in high-dimensional flows [4, 41].

To alleviate the computational cost of evaluating the flow Jacobian at inference, Flow Perturbation (FP) [43] provides an unbiased estimator for the Jacobian determinant when the directional derivative is evaluated exactly:

$$\hat{\mathcal{J}}_{\text{FP}} = \left\| \frac{\partial \mathbf{f}^{-1}}{\partial \mathbf{x}} \epsilon \right\|^{-D}, \quad \epsilon \sim \mathbb{S}^{D-1}, \tag{5}$$

where ϵ is sampled uniformly from the unit sphere in D dimensions. For exact inverse derivatives, $\mathbb{E}[\hat{\mathcal{J}}_{\text{FP}}] = |\det(\partial \mathbf{f}/\partial \mathbf{z})|$.

4 Method

4.1 Flow Perturbation++ estimator

As Figure 1 illustrates, numerical integration decomposes the flow $\mathbf{f}$ into T steps,

$$\mathbf{f} = \mathbf{f}_1 \circ \cdots \circ \mathbf{f}_T,$$
$$\left|\det\left(\frac{\partial \mathbf{f}}{\partial \mathbf{z}}\right)\right| = \prod_{k=1}^{T} \left|\det(J_k)\right|. \tag{6}$$

Here J_k is the Jacobian of $\mathbf{f}_k$ at the corresponding point on the trajectory. FP++ applies a separate perturbation to each inverse step and multiplies the resulting determinant estimates,

$$\hat{\mathcal{J}}_{\text{FP++}} = \prod_{k=1}^{T} \|J_k^{-1}\epsilon_k\|^{-D}, \quad \epsilon_k \sim \mathbb{S}^{D-1}, \tag{7}$$

With independent directions across steps, the product is unbiased for exact inverse-step derivatives:

$$\mathbb{E}[\hat{\mathcal{J}}_{\text{FP++}}] = \left|\det \frac{\partial \mathbf{f}}{\partial \mathbf{z}}\right|. \tag{8}$$

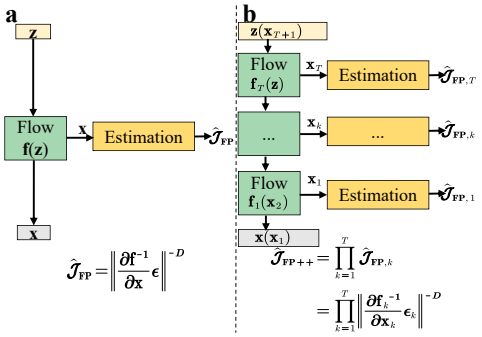


Figure 1: **Flow Perturbation vs FP++.** (a) Single-step FP estimates the full-flow determinant in one shot, which can lead to high variance. (b) FP++ decomposes flow into T steps and applies determinant estimation sequentially from latent $\mathbf{z} = \mathbf{x}_{T+1}$ to $\mathbf{x} = \mathbf{x}_1$. The product of per-step estimates yields the FP++ determinant estimator.

Appendix B provides a formal proof.

In practice, the inverse-JVP is approximated via central finite differences:

$$J_k^{-1}\epsilon_k \approx \frac{\mathbf{f}_k^{-1}(x_k + \delta\epsilon_k) - \mathbf{f}_k^{-1}(x_k - \delta\epsilon_k)}{2\delta}. \tag{9}$$

Here δ is the perturbation size. Both inverse evaluations use the same direction ϵ_k, avoiding explicit construction of J_k, while fresh spherical directions are sampled across steps. This finite-difference approximation introduces numerical error and is distinct from the exact-derivative unbiasedness statement above. Convergence and perturbation-size sensitivity are examined in Appendices H.2 and H.3, with inverse consistency discussed in Appendix B.3.

Variance reduction. For a fixed trajectory, independent probes eliminate cross-step covariance in the log-domain sum. FP++ has lower log-domain variance when one-step variances match and FP has positive aggregate cross-step covariance. Appendix C gives a small-step stretch-alignment condition for the latter; Appendix H.5 evaluates determinant and importance-weight variance.

4.2 Integration with sequential Monte Carlo

In high-dimensional systems, importance weights often concentrate on only a few samples. We use SMC over N levels, $0 = \beta_0 < \cdots < \beta_{N-1} = 1$, with the ideal exact-Jacobian geometric bridge

$$\pi_n(\mathbf{x}) \propto p_B(\mathbf{x})^{\beta_n} q_f(\mathbf{x})^{1-\beta_n}, \tag{10}$$

and update incremental weights via

$$\omega_n(\mathbf{x}) = \omega_{n-1}(\mathbf{x}) \exp[-(\beta_n - \beta_{n-1}) W(\mathbf{x})], \tag{11}$$

where q_f is the flow-induced density and $W(\mathbf{x}) = \log q_f(\mathbf{x}) - \log p_B(\mathbf{x})$ up to constants. FP++ supplies the determinant correction in $\log q_f$. We resample when $\mathrm{ESS} = (\sum_j (\omega_n^{(j)})^2)^{-1}$ falls below a threshold.

Algorithm 1 SMC with Flow Perturbation++

Input: $q_f, p_B, M, \{\beta_n\}_{n=0}^{N-1}, \tau, L$
Init: $\mathbf{x}_0^{(i)} \sim q_f$, $\omega_0^{(i)} = 1/M$
for $n = 1$ to $N-1$:

1. Compute the FP++ determinant correction entering $W^{(i)}$
2. Update and normalize weights:
$$\omega_n^{(i)} \propto \omega_{n-1}^{(i)} e^{-(\beta_n - \beta_{n-1}) W^{(i)}}$$
3. If $\mathrm{ESS}_n < \tau M$, resample and set $\omega_n^{(i)} = 1/M$
4. Apply L π_{β_n}-invariant K_{β_n} moves

Output: $\{\mathbf{x}_{N-1}^{(i)}, \omega_{N-1}^{(i)}\}$

The full FP++–SMC procedure is summarized in Algorithm 1. With stochastic determinant estimates, the intermediate targets are extended-state targets (Appendix A), not this exact-Jacobian bridge; the ideal endpoint marginal remains the Boltzmann target. Finite-particle and numerical approximations remain.

4.3 Computational complexity comparison

We compare the computational cost of baseline Jacobian-related estimators with the proposed FP++ (multi-step) approach. Baselines include single-step Flow Perturbation (FP) [43], the Hutchinson trace estimator [21], and the exact brute-force Jacobian (BFJacob).

Table 1 summarizes their computational characteristics. All methods require one forward ODE integration to generate the trajectory $\mathbf{x} = f(\mathbf{z})$; additional cost arises from estimating the Jacobian-related determinant correction.

Cost unit. We measure computational complexity in terms of *ODE-equivalent passes*, where one pass corresponds to a full trajectory integration (forward or backward along the flow).

- **BFJacob:** Computes the full $D \times D$ Jacobian via D vector–Jacobian products (VJPs). Each VJP requires one backward pass along the flow. The total cost is therefore $1 + D$ passes.
- **Hutchinson estimator:** Approximates the trace using N random probe vectors. Each probe requires one VJP, resulting in a total cost of $1 + N$ passes.
- **FP / FP++:** We estimate the Jacobian-related term using finite-difference evaluations of the inverse flow. Each probe requires two evaluations of the inverse dynamics along the trajectory. Therefore, using N probes leads to a total cost of $1 + 2N$ passes.

Table 1: Computational cost comparison of Jacobian-related estimators. Cost is measured in ODE-equivalent passes. N denotes the number of probes; in our experiments, we set $N = 1$ for FP / FP++.

Method	Jacobian Calc.	Biased	Cost (# passes)
FP / FP++	No	No	$1 + 2N$
BFJacob	Yes	No	$1 + D$
Hutchinson	No	Yes	$1 + N$

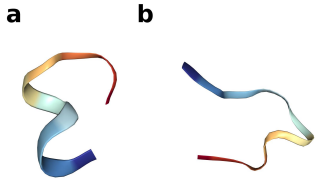


Figure 2: Representative CNL025 configurations at 300 K: (a) partially α-helical; (b) folded β-hairpin.

5 Experiments and results

We evaluate the FP++-SMC framework on two representative high-dimensional systems: a synthetic Gaussian Mixture Model (GMM), and the all-atom Chignolin peptide.

5.1 Experimental setup

5.1.1 Test systems

Gaussian mixture model (GMM). We consider a high-dimensional two-component anisotropic Gaussian mixture as a benchmark system. Training samples are drawn uniformly from the mixture, while the target Boltzmann distribution assigns asymmetric component weights $(0.25,\ 0.75)$, making it challenging for models to fully match the target distribution. This bimodal system is structurally simple, yet poses a nontrivial sampling challenge due to metastability between modes, and admits an analytically tractable reference solution, making it an ideal platform for evaluating the performance of FP++. Full system specifications are provided in Appendix D.

Chignolin mutant CNL025. The 10-residue Chignolin mutant CNL025 [46, 37, 36] provides a compact and challenging biomolecular test case. We adopt the CHARMM22 force field [34] with the OBC2 implicit solvent model [42]. Training configurations at 300 K are extracted from replica-exchange molecular dynamics data released by the Noé group.[2] Representative conformations are shown in Fig. 2.

5.1.2 Flow models

For all experiments, we use the pretrained continuous-time flow models released in [43] without further training. The models were trained with the v-prediction objective [47], which improves numerical stability in high dimensions, with the neural network representing the time-dependent velocity field $v_\theta(\mathbf{x}, t)$ that defines the probability flow. Full architecture and training details are reported in [43].

For the GMM benchmark, a lightweight multilayer perceptron is sufficient to model the low-complexity vector field associated with the mixture structure. For Chignolin, we employ a Transformer-based architecture [56, 29] to parameterize $v_\theta(\mathbf{x}, t)$, enabling the model to capture long-range geometric and many-body correlations in high-dimensional biomolecular configurations.

5.1.3 Compared Jacobian estimators

We evaluate estimators under identical SMC pipelines, all using a 20-step RK4 ODE solver:

- **FP / FP++ (single-probe,** $N = 1$**)** [43], perturbation-based estimators using one finite-difference probe.
- **Hutchinson trace estimators** [21], using Gaussian or Rademacher probes, denoted as 'Gn' and 'Rn' respectively, with $n \in \{1, 2, 10\}$.
- **Brute-force Jacobian (BFJacob)**, an exact but computationally expensive method via automatic differentiation.

5.1.4 Evaluation metrics

We report several standard metrics to quantify estimator accuracy and sampling performance.

- **Energy Distribution.** We compare empirical SMC energy histograms with the reference Boltzmann distribution.
- **Computational Cost.** We measure the approximate wall-clock time required for one full SMC run.
- **Reaction-Coordinate Occupancy.** We compute occupancy along a selected reaction coordinate to evaluate recovery of the free-energy landscape.

[2] `http://ftp.mi.fu-berlin.de/pub/cmbdata/bgmol/datasets/chignolin/ChignolinOBC2PT.tgz`

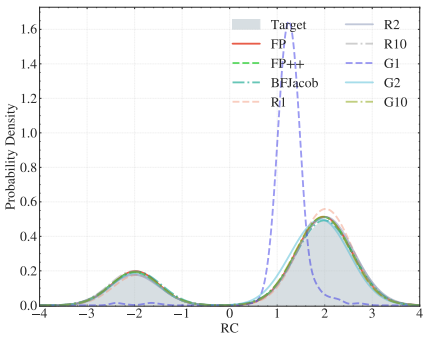


Figure 3: Final SMC sample distribution along the reaction coordinate for the 10D GMM. The solid blue curve shows the target distribution obtained via direct sampling.

Table 2: Performance of baseline and proposed estimators on the 10D GMM. The modal weight reports the fraction of SMC particles located in the first Gaussian component.

Estimator	Mean	Std	# Preserved	Time
Hutchinson G1	0.0189	0.018	131	2.5m
Hutchinson G2	0.2278	0.013	828	3.0m
Hutchinson G10	0.2454	0.011	943	3.0m
Hutchinson R1	0.2289	0.017	972	2.5m
Hutchinson R2	0.2391	0.010	971	3.0m
Hutchinson R10	0.2503	0.016	971	3.0m
BF Jacobian	0.2497	0.019	978	5.5m
FP (single-step)	0.2499	0.015	862	2.5m
FP++ (multi-step)	0.2503	0.011	957	2.5m

- **Remaining Ancestor Count.** We track the number of surviving ancestors after resampling [8], a diagnostic indicator of particle degeneracy.

In addition, we include supplementary evaluations in Appendix H, including variance analysis of the log-domain Jacobian estimator across methods and dimensionalities, convergence studies of FP++ and Hutchinson-based estimators, stability evaluations on Chignolin, sensitivity analyses with respect to finite-difference step size, direct determinant/weight variance and ESS measurements, and an additional validation on a molecular system, alanine dipeptide from Transferable Boltzmann Generators (TBG) [24].

5.2 Results

Before presenting the specific tasks, we briefly discuss the numerical robustness of the proposed FP++ estimator. We conduct sensitivity and stability analyses (see Appendix H) to assess its reliability in importance sampling. Empirically, FP++ achieves a variance level comparable to the brute-force Jacobian even with a single probe, and remains stable across dimensionalities ranging from 10D to 1000D. Our sensitivity study further shows that setting the finite-difference step size $\delta = 0.001$ yields accurate and stable estimates consistent with the JVP reference across all tested systems. In addition, repeated independent runs show that the estimator mean converges to the brute-force Jacobian reference as the number of runs M increases. These results confirm that FP++ provides a stable and low-variance estimator suitable for the subsequent Boltzmann sampling experiments.

5.2.1 Low-dimensional GMM (10D)

We assess estimator performance on a 10-dimensional Gaussian Mixture Model using SMC with 1,000 particles across 10 independent runs. Accuracy is measured by the fraction of particles assigned to the first Gaussian component. As shown in Table 2, both the FP baseline and FP++ accurately recover the target modal probability (25%), matching the ground truth. Hutchinson estimators exhibit noticeable bias when using only a few random probe vectors ($n = 1, 2$), with accuracy improving as the number of probe noises increases. The brute-force Jacobian ("BF Jacobian"), using results reported in [43], provides high accuracy but at substantially greater computational cost. The reaction-coordinate distributions in Fig. 3 further demonstrate that FP++ yields stable and accurate reconstruction of the target distribution.

5.2.2 1000 dimensional GMM

We further benchmark all estimators on a substantially more challenging 1000-dimensional GMM. For FP, FP++, and Hutchinson variants, SMC is performed with 1,000 particles and ten independent runs, using 2,000 annealing steps with 10 MCMC transitions per step. Due to its prohibitive cost, the Brute-Force Jacobian baseline (BFJacob) is evaluated using 100 particles in a single run, using results reported in [43].

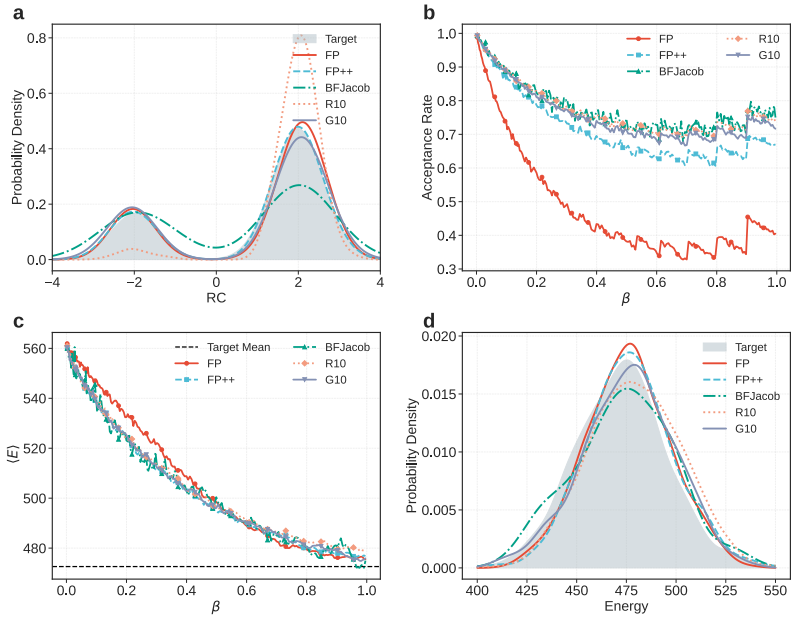


Figure 4: SMC results for the 1000-dimensional GMM. (a) Probability distribution of final SMC samples along the reaction coordinate. (b) Monte Carlo acceptance rate as a function of β. (c) Average sample energy as a function of β. (d) Final energy distribution. The target energy and distribution (solid line) are obtained by direct sampling from the target GMM. All results are derived from a single SMC run.

Table 3: Performance of Jacobian estimators on the 1000-dimensional GMM. Modal weights report the fraction of particles assigned to the first Gaussian component.

Estimator	Mean	Std	# Preserved	Time
G1	0.534	0.073	154	0.7 h
G2	0.377	0.023	222	1.1 h
G10	0.277	0.022	360	5.1 h
R1	0	0	328	0.7 h
R2	0	0	361	1.1 h
R10	0.044	0.011	409	5.1 h
BF Jacobian	0.400	–	45	31 h
FP	0.209	0.083	59	1.1 h
FP++	**0.256**	0.027	312	1.1 h

Table 4: Performance of different Jacobian estimators on the Chignolin mutant. The table reports the fraction of particles in the β-hairpin basin (Fig. 5), along with preserved lineages and computational cost. Mean, Std, and preserved lineages summarize ten runs; time is per run. Target value is 0.385; base flow yields 0.311.

Method	Mean	Std	# Preserved	Time
R2	0.2330	0.0168	1529	30.4 h
R10	0.2457	0.0369	1541	100 h
G2	0.2136	0.0685	1316	30.4 h
G10	0.2303	0.0394	1458	100 h
FP	0.4450	0.0510	1433	11.1 h
FP++	**0.3658**	**0.0146**	1474	11.1 h

Table 3 reports the estimated probability of the first mode (ground truth 25%), lineage preservation, and computational cost. Among all methods, **FP++ best recovers the target modal probability**, yielding 0.256 ± 0.027 across ten runs while preserving a large number of lineages (312 on average). Notably, FP++ achieves this accuracy at essentially the same computational cost as the single-step FP estimator.

Using 10 Gaussian Hutchinson probes (G10) attains the next-best performance (0.277 ± 0.022), but requires roughly five times the computational cost. The single-step FP estimator produces reasonable estimates (0.209 ± 0.083) with substantially higher variance. All other Hutchinson variants suffer from significant bias or mode collapse, accompanied by severe lineage degeneracy. BFJacob underperforms at this scale due to the limited particle budget; extrapolating to 1,000 particles would increase its computational cost by roughly 300× compared with FP++, making it entirely impractical.

Figure 4 shows final sample and energy distributions (a,d) and acceptance rates and average energies across the annealing schedule (b,c). FP++ consistently matches or exceeds the accuracy of both

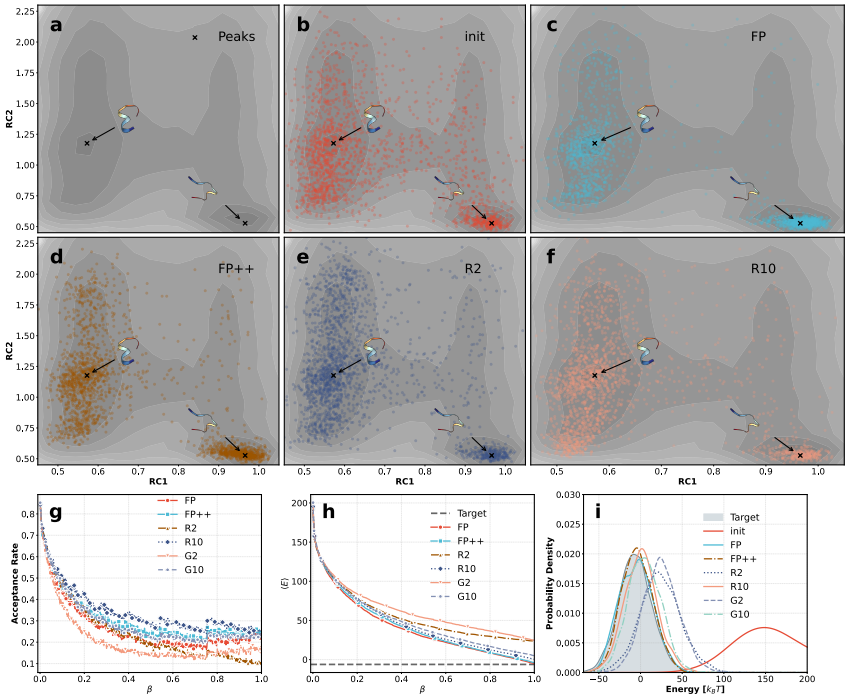


Figure 5: SMC results for the Chignolin mutant. (a–f) Sample distributions projected onto the reaction-coordinate (RC) plane: (a) Target distribution derived from the training data; (b) Samples generated directly by the base flow model; (c) Samples produced by SMC using the single-step FP estimator; (d) Samples produced by SMC using the FP++ estimator; (e) Samples produced by SMC with the Hutchinson estimator (10 Rademacher vectors); (f) Samples produced by SMC with the Hutchinson estimator (10 Gaussian vectors). (g) Monte Carlo acceptance rates as functions of β; (h) Average energy $\langle E \rangle$ of intermediate SMC samples; (i) Energy distributions at the end of SMC compared to the target distribution and the base flow ("init"). All SMC results are derived from a single run.

G10 and BFJacob while using only a fraction of their computational budget. Partial coordinate updates induce stepwise acceptance-rate fluctuations, with FP exhibiting the highest variance and lowest acceptance rates. FP++ reduces this variance by per-step Jacobian decomposition, resulting in consistently higher acceptance rates.

5.2.3 Chignolin mutant

For the Chignolin mutant, we conducted SMC simulations using the FP, FP++, and Hutchinson estimators. Each simulation employed 2,000 particles, 600 intermediate distributions, and 2 MCMC steps per level. Resampling was performed using the KL Reshuffling scheme [27]. The Brute-Force Jacobian baseline (BFJacob) was omitted due to its prohibitive computational cost: even a single run with a reduced number of particles would already require orders of magnitude more resources than FP++ or Hutchinson estimators, making it infeasible for this system.

Table 4 reports the fraction of particles occupying the right (folded) basin, with a target occupancy of 0.385 and a base-flow value of 0.311. Among all tested estimators, **FP++ achieves the closest agreement with the target**, yielding 0.3658 ± 0.0146 across ten runs (range 0.3455–0.3875) while preserving 1,474 particles on average. The single-step FP estimator overshoots the target at 0.4450 ± 0.0510 (ten-run range 0.3320–0.4830) under the same computational budget. Hutchinson-based estimators systematically underestimate basin occupancy, even with 10 Gaussian or Rademacher probes. Variants with fewer probes (G2/R2) remain less accurate than FP++ under this matched SMC setting; complete ten-run statistics and ranges are reported in Appendix H.4, Table 9.

Although finite-difference approximations incur a comparable number of model evaluations to low-probe Hutchinson estimators (e.g., G2/R2), they can be substantially faster in practice for Transformer-

based flows. As shown in Table 4, finite-difference–based FP and FP++ achieve significantly lower wall-clock times than Hutchinson-type estimators under comparable or even larger computational budgets. This behavior arises from the use of self-attention and normalization layers: Jacobian–vector products require backpropagation through these components, whose computational cost is dominated by dense attention operations and softmax derivatives. Such backward passes are well known to be considerably more expensive than forward evaluations [56, 30]. In contrast, finite-difference estimators only involve multiple forward passes, which are heavily optimized on modern GPUs via fused kernels and specialized attention implementations. For consistency, Appendix F reports runtimes with all methods implemented using VJP or JVP, including FP and FP++.

Figure 5 illustrates the projected particle distributions along the reaction-coordinate plane. Both FP and FP++ faithfully recover the bimodal target distribution, with negligible leakage into the low-probability corridor separating the α-helical and β-hairpin basins. In contrast, Hutchinson-based and base-flow samples display substantial spurious occupancy between basins.

Energy histograms (Figure 5(i)) further confirm that FP and FP++ generate high-fidelity ensembles compared to Hutchinson estimators or direct base-flow samples. Acceptance rates along the annealing schedule (Figure 5(g)) remain smooth, consistent with largely fixed-coordinate updates, and FP++ exhibits a slight improvement over FP, reflecting reduced variance through the multi-step decomposition.

Complementary diagnostics for representative Chignolin runs with 2,000 particles are reported in Appendix H.4 (Table 10). FP++ has the highest pre-resampling importance ESS/N (0.003499) and a torsion-W2 (2.7349) close to R10 (2.6891); FP has lower Energy-W2 (1.6512 versus 2.7330). R2/R10 retain more ancestry but underestimate folded-basin occupancy. These weight, distributional, and genealogical metrics reveal trade-offs rather than uniform improvement.

For reference, conventional MD trajectories often become trapped in single metastable states, underscoring the challenges of effectively sampling the Chignolin mutant using standard simulation approaches (Appendix G).

6 Conclusion

We have introduced **Flow Perturbation++ (FP++)**, a theoretically grounded estimator of the flow Jacobian determinant that leverages the semigroup property of flow maps to decompose the global Jacobian into per-step determinants. It is determinant-unbiased for exact inverse-step derivatives, with variance reduction under the conditions in Appendix C. FP++ serves as a practical and scalable alternative to Hutchinson-style trace estimators [21]. When integrated with sequential Monte Carlo (SMC), FP++ enables accurate Boltzmann reweighting and reliable thermodynamic statistics across both synthetic high-dimensional Gaussian mixtures and realistic biomolecular systems such as the Chignolin mutant. Our experiments demonstrate that FP++ not only improves estimator accuracy and stability compared to both the original FP and Hutchinson estimators, but also substantially reduces computational cost relative to conventional brute-force Jacobian evaluation.

Limitations. Finite differences and reverse integration introduce numerical errors (Appendices H.3 and B.3), and variance reduction is conditional rather than universal. Exact Jacobians remain preferable when numerical approximations cannot be validated. FP++ complements sampler pipelines; end-to-end comparisons with FALCON or related samplers remain future work.

Acknowledgments and Disclosure of Funding

This work was supported by the Opening Project of the State Key Laboratory of Information Photonics and Optical Communications (Grant No. IPOC2025ZT02), the National Natural Science Foundation of China (Grant No. 22473016), the Beijing Natural Science Foundation (Grant No. L258008), and the Fundamental Research Funds for Beijing University of Posts and Telecommunications (Grant No. 2025AI4S01). We are thankful to the anonymous reviewers for their helpful comments. We thank the Shanghai Academy of AI for Science and Fudan University for providing computational resources through the CFFF platform during the rebuttal period. The authors declare no competing interests.
Software and Data. Implementations of the methods and scripts for data generation and evaluation are publicly available at `https://github.com/XinPeng76/Flow_Perturbation_pp`.

References

[1] Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J. Ballard, Joshua Bambrick, Sebastian W. Bodenstein, David A. Evans, Chia-Chun Hung, Michael O'Neill, David Reiman, Kathryn Tunyasuvunakool, Zachary Wu, Akvilė Žemgulytė, Eirini Arvaniti, Charles Beattie, Ottavia Bertolli, Alex Bridgland, Alexey Cherepanov, Miles Congreve, Alexander I. Cowen-Rivers, Andrew Cowie, Michael Figurnov, Fabian B. Fuchs, Hannah Gladman, Rishub Jain, Yousuf A. Khan, Caroline M. R. Low, Kuba Perlin, Anna Potapenko, Pascal Savy, Sukhdeep Singh, Adrian Stecula, Ashok Thillaisundaram, Catherine Tong, Sergei Yakneen, Ellen D. Zhong, Michal Zielinski, Augustin Žídek, Victor Bapst, Pushmeet Kohli, Max Jaderberg, Demis Hassabis, and John M. Jumper. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, 2024. doi: 10.1038/s41586-024-07487-w. URL https://doi.org/10.1038/s41586-024-07487-w.

[2] Michael S Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*, 2023.

[3] Michael Arbel, Alexander G. D. G. Matthews, and Arnaud Doucet. Annealed flow transport monte carlo. In *Proceedings of the 38th International Conference on Machine Learning*, Proceedings of Machine Learning Research, 18–24 Jul 2021.

[4] Jens Behrmann, Will Grathwohl, Ricky T. Q. Chen, David Duvenaud, and Joern-Henrik Jacobsen. Invertible residual networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 573–582, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL http://proceedings.mlr.press/v97/behrmann19a.html.

[5] Yoshua Bengio, Salem Lahlou, Tristan Deleu, Edward J. Hu, Mo Tiwari, and Emmanuel Bengio. Gflownet foundations. *Journal of Machine Learning Research*, 24(210):1–55, 2023. URL http://jmlr.org/papers/v24/22-0364.html.

[6] Chen-Hao Chao, Chien Feng, Wei-Fang Sun, Cheng-Kuang Lee, Simon See, and Chun-Yi Lee. Maximum entropy reinforcement learning via energy-based normalizing flow. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=lhlIUxD5eE.

[7] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, volume 31, 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/69386f6bb1dfed68692a24c8686939b9-Paper.pdf.

[8] Pierre Del Moral, Arnaud Doucet, and Ajay Jasra. Sequential monte carlo samplers. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 68(3):411–436, 05 2006. ISSN 1369-7412. doi: 10.1111/j.1467-9868.2006.00553.x. URL https://doi.org/10.1111/j.1467-9868.2006.00553.x.

[9] Christoph Dellago, Peter G. Bolhuis, Félix S. Csajka, and David Chandler. Transition path sampling and the calculation of rate constants. *The Journal of Chemical Physics*, 108(5):1964–1977, 02 1998. ISSN 0021-9606. doi: 10.1063/1.475562. URL https://doi.org/10.1063/1.475562.

[10] Niclas Dern, Lennart Redl, Sebastian Pfister, Marcel Kollovieh, David Lüdke, and Stephan Günnemann. Energy-weighted flow matching: Unlocking continuous normalizing flows for efficient and scalable boltzmann sampling, 2026. URL https://arxiv.org/abs/2509.03726.

[11] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation, 2015. URL https://arxiv.org/abs/1410.8516.

[12] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=HkpbnH9lx.

[13] Ezra Erives, Bowen Jing, and Tommi Jaakkola. Verlet flows: Exact-likelihood integrators for flow-based generative models. In *ICLR 2024 Workshop on AI4DifferentialEquations In Science*, 2024. URL https://openreview.net/forum?id=JkYOOK5ulR.

[14] Hans Frauenfelder, Stephen G. Sligar, and Peter G. Wolynes. The energy landscapes and motions of proteins. *Science*, 254(5038):1598–1603, 1991.

[15] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1352–1361. JMLR.org, 2017.

[16] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1861–1870. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/haarnoja18b.html.

[17] Jiajun He, José Miguel Hernández-Lobato, Yuanqi Du, and Francisco Vargas. RNE: plug-and-play diffusion inference-time control and energy-based training. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=fkf7tlkN8q.

[18] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851, 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

[19] Emiel Hoogeboom, Víctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3D. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 8867–8887. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/hoogeboom22a.html.

[20] 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. doi: 10.1143/JPSJ.65.1604. URL https://doi.org/10.1143/JPSJ.65.1604.

[21] M. F. Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics - Simulation and Computation*, 18(2):1059–1076, 1989.

[22] Vineet Jain, Tara Akhound-Sadegh, and Siamak Ravanbakhsh. Sampling from energy-based policies using diffusion, 2025. URL https://arxiv.org/abs/2410.01312.

[23] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, volume 31, 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/d139db6a236200b21cc7f752979132d0-Paper.pdf.

[24] Leon Klein and Frank Noé. Transferable boltzmann generators. *Advances in Neural Information Processing Systems*, 37:45281–45314, 2024. URL https://arxiv.org/abs/2406.14426.

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

[26] Jonas Köhler, Leon Klein, and Frank Noe. Equivariant flows: Exact likelihood generative learning for symmetric densities. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5361–5370. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/kohler20a.html.

[27] Oskar Kviman, Hazal Koptagel, Harald Melin, and Jens Lagergren. Statistical distance based deterministic offspring selection in smc methods. *arXiv preprint arXiv:2212.12290*, 2022.

[28] Alessandro Laio and Michele Parrinello. Escaping free-energy minima. *Proceedings of the National Academy of Sciences*, 99(20):12562–12566, 2002. doi: 10.1073/pnas.202427399. URL https://doi.org/10.1073/pnas.202427399.

[29] Junlong Li, Yiheng Xu, Tengchao Lv, Lei Cui, Cha Zhang, and Furu Wei. Dit: Self-supervised pre-training for document image transformer. In *Proceedings of the 30th ACM International Conference on Multimedia*, MM '22, page 3530–3539, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450392037. doi: 10.1145/3503161.3547911. URL https://doi.org/10.1145/3503161.3547911.

[30] Yingyu Liang, Heshan Liu, Zhenmei Shi, Zhao Song, Zhuoyan Xu, and Junze Yin. Conv-basis: A new paradigm for efficient attention inference and gradient computation in transformers, 2025. URL https://openreview.net/forum?id=yD7oAhFEtD.

[31] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

[32] Jun S Liu and Rong Chen. Sequential monte carlo methods for dynamic systems. *Journal of the American statistical association*, 93(443):1032–1044, 1998.

[33] Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*, 2023. URL https://openreview.net/forum?id=XVjTT1nw5z.

[34] Alexander D. MacKerell Jr., Nilesh Banavali, and Nicolas Foloppe. Development and current status of the charmm force field for nucleic acids. *Biopolymers*, 56(4):257–265, 2000. doi: https://doi.org/10.1002/1097-0282(2000)56:4<257::AID-BIP10029>3.0.CO;2-W. URL https://doi.org/10.1002/1097-0282(2000)56:4<257::AID-BIP10029>3.0.CO;2-W.

[35] Nikolay Malkin, Moksh Jain, Emmanuel Bengio, Chen Sun, and Yoshua Bengio. Trajectory balance: improved credit assignment in gflownets. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

[36] Yutaka Maruyama and Ayori Mitsutake. Analysis of structural stability of chignolin. *The Journal of Physical Chemistry B*, 122(14):3801–3814, 04 2018. doi: 10.1021/acs.jpcb.8b00288. URL https://doi.org/10.1021/acs.jpcb.8b00288.

[37] Keri A. McKiernan, Brooke E. Husic, and Vijay S. Pande. Modeling the mechanism of cln025 beta-hairpin formation. *The Journal of Chemical Physics*, 147(10):104107, 09 2017. ISSN 0021-9606. doi: 10.1063/1.4993207. URL https://doi.org/10.1063/1.4993207.

[38] Safa Messaoud, Billel Mokeddem, Zhenghai Xue, Linsey Pang, Bo An, Haipeng Chen, and Sanjay Chawla. S2AC: Energy-based reinforcement learning with stein soft actor critic. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=rAHcTCMaLc.

[39] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953. doi: 10.1063/1.1699114. URL https://doi.org/10.1063/1.1699114.

[40] Laurence Illing Midgley, Vincent Stimper, Gregor N.C. Simm, Bernhard Schölkopf, and José Miguel Hernández-Lobato. Flow annealed importance sampling bootstrap. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2208.01893.

[41] Frank Noé, Simon Olsson, Jonas Köhler, and Hao Wu. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science*, 365(6457):1–11, 2019. doi: 10.1126/science.aaw1147.

[42] Alexey Onufriev, Donald Bashford, and David A. Case. Exploring protein native states and large-scale conformational changes with a modified generalized born model. *Proteins: Structure, Function, and Bioinformatics*, 55(2):383–394, 2004. doi: https://doi.org/10.1002/prot.20033. URL https://doi.org/10.1002/prot.20033.

[43] Xin Peng and Ang Gao. Flow perturbation to accelerate boltzmann sampling. *Nature Communications*, 16(1):6604, 2025.

[44] Danyal Rehman, Tara Akhound-Sadegh, Artem Gazizov, Yoshua Bengio, and Alexander Tong. FALCON: Few-step accurate likelihoods for continuous flows. *arXiv preprint arXiv:2512.09914*, 2025. URL https://arxiv.org/abs/2512.09914.

[45] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on Machine Learning - Volume 37*, pages 1530–1538, 2015.

[46] Alex Rodriguez, Pol Mokoema, Francesc Corcho, Khrisna Bisetty, and Juan J. Perez. Computational study of the free energy landscape of the miniprotein cln025 in explicit and implicit solvent. *The Journal of Physical Chemistry B*, 115(6):1440–1449, 02 2011. doi: 10.1021/jp106475c. URL https://doi.org/10.1021/jp106475c.

[47] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=TIdIXIpzhoI.

[48] Wenjie Shi, Shiji Song, and Cheng Wu. Soft policy gradient method for maximum entropy deep reinforcement learning, 2019. URL https://arxiv.org/abs/1909.03198.

[49] Marta Skreta, Tara Akhound-Sadegh, Viktor Ohanesian, Roberto Bondesan, Alan Aspuru-Guzik, Arnaud Doucet, Rob Brekelmans, Alexander Tong, and Kirill Neklyudov. Feynman-kac correctors in diffusion: Annealing, guidance, and product of experts. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=Vhc0KrcqWu.

[50] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=PxTIG12RRHS.

[51] Matthias Speich, Carsten F. Dormann, and Florian Hartig. Sequential monte-carlo algorithms for bayesian model calibration – a review and method comparison. *Ecological Modelling*, 455:109608, 2021. ISSN 0304-3800. doi: https://doi.org/10.1016/j.ecolmodel.2021.109608. URL https://www.sciencedirect.com/science/article/pii/S0304380021001708.

[52] Robert H. Swendsen and Jian-Sheng Wang. Replica monte carlo simulation of spin-glasses. *Phys. Rev. Lett.*, 57:2607–2609, Nov 1986. doi: 10.1103/PhysRevLett.57.2607. URL https://link.aps.org/doi/10.1103/PhysRevLett.57.2607.

[53] Charlie B. Tan, Avishek Joey Bose, Chen Lin, Leon Klein, Michael M. Bronstein, and Alexander Tong. Scalable equilibrium sampling with sequential boltzmann generators. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pages 58467–58498. PMLR, 2025. URL https://proceedings.mlr.press/v267/tan25a.html.

[54] Charlie B. Tan, Majdi Hassan, Leon Klein, Saifuddin Syed, Dominique Beaini, Michael M. Bronstein, Alexander Tong, and Kirill Neklyudov. Amortized sampling with transferable normalizing flows. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/8780dbea239de8fd1401f9ec6721608d-Abstract-Conference.html.

[55] G.M. Torrie and J.P. Valleau. Nonphysical sampling distributions in monte carlo free-energy estimation: Umbrella sampling. *Journal of Computational Physics*, 23(2):187–199, 1977. ISSN 0021-9991. doi: https://doi.org/10.1016/0021-9991(77)90121-8. URL https://www.sciencedirect.com/science/article/pii/0021999177901218.

[56] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[57] Loup Verlet. Computer "experiments" on classical fluids. i. thermodynamical properties of lennard-jones molecules. *Phys. Rev.*, 159:98–103, Jul 1967. doi: 10.1103/PhysRev.159.98. URL https://link.aps.org/doi/10.1103/PhysRev.159.98.

[58] L. Wang, G. Aarts, and K. Zhou. Diffusion models as stochastic quantization in lattice field theory. *Journal of High Energy Physics*, 2024(5):60, 2024. doi: 10.1007/JHEP05(2024)060. URL https://doi.org/10.1007/JHEP05(2024)060.

[59] Yuyang Wang, Ahmed A. A. Elhag, Navdeep Jaitly, Joshua M. Susskind, and Miguel Ángel Bautista. Generating molecular conformer fields, 2024. URL https://openreview.net/forum?id=XSwxy3bojg.

[60] Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. Geodiff: A geometric diffusion model for molecular conformation generation. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=PzcvxEMzvQC.

[61] Shuxin Zheng, Jiyan He, Chang Liu, Yu Shi, Ziheng Lu, Weitao Feng, Fusong Ju, Jiaxi Wang, Jianwei Zhu, Yaosen Min, He Zhang, Shidi Tang, Hongxia Hao, Peiran Jin, Chi Chen, Frank Noé, Haiguang Liu, and Tie-Yan Liu. Predicting equilibrium distributions for molecular systems with deep learning. *Nature Machine Intelligence*, 2024. doi: 10.1038/s42256-024-00837-3. URL https://doi.org/10.1038/s42256-024-00837-3.

[62] Xuekai Zhu, Daixuan Cheng, Dinghuai Zhang, Hengli Li, Kaiyan Zhang, Che Jiang, Youbang Sun, Ermo Hua, Yuxin Zuo, Xingtai Lv, Qizheng Zhang, Lin Chen, Fanghao Shao, Bo Xue, Yunchong Song, Zhenjie Yang, Ganqu Cui, Ning Ding, Jianfeng Gao, Xiaodong Liu, Bowen Zhou, Hongyuan Mei, and Zhouhan Lin. Flowrl: Matching reward distributions for llm reasoning, 2025. URL https://arxiv.org/abs/2509.15207.

A Detailed sequential Monte Carlo protocol

This section describes the SMC procedure used with the different Jacobian estimators. The particle count, annealing schedule, mutation kernels, and resampling rule are matched across methods. For stochastic determinant estimators, the target distribution is formulated on an extended state space that includes the probes.

Intermediate targets and weights. Let $q(\mathbf{z})$ be the latent density, $\mathbf{x} = \mathbf{f}(\mathbf{z})$, and $p_B(\mathbf{x}) \propto \exp[-E(\mathbf{x})]$. With an exact Jacobian, the usual geometric bridge in configuration space is $\pi_\beta(\mathbf{x}) \propto q_f(\mathbf{x})^{1-\beta} p_B(\mathbf{x})^\beta$. For FP++, let ξ denote the complete collection of raw auxiliary variables and let $\rho(\xi)$ be their joint density. Each spherical probe is represented as $\epsilon = \xi/\|\xi\|$ with $\xi \sim \mathcal{N}(0, I)$, so refreshing Gaussian coordinates preserves the uniform spherical direction after normalization. We write $\widehat{\mathcal{J}}(\mathbf{z}, \xi)$ for the resulting positive determinant estimator and define

$$\widehat{\mathcal{J}}(\mathbf{z}, \xi) = \exp[\Delta S(\mathbf{z}, \xi)], \qquad R(\mathbf{z}, \xi) = \frac{p_B(\mathbf{f}(\mathbf{z}))}{q(\mathbf{z})} \widehat{\mathcal{J}}(\mathbf{z}, \xi). \tag{11}$$

The corresponding extended targets are

$$\widetilde{\pi}_\beta(\mathbf{z}, \xi) \propto q(\mathbf{z}) \rho(\xi) R(\mathbf{z}, \xi)^\beta. \tag{12}$$

At $\beta = 0$, Eq. (12) is the latent proposal augmented by independent probes. For $0 < \beta < 1$, its configuration marginal generally differs from $q_f^{1-\beta} p_B^\beta$ because $\mathbb{E}[\widehat{\mathcal{J}}^{\,\beta}] \neq \mathbb{E}[\widehat{\mathcal{J}}]^\beta$ in general. The endpoint marginal is characterized by the following proposition.

Proposition A.1 (Endpoint marginal of the ideal extended target)**.** *Assume that* $\mathbf{f}$ *is a diffeomorphism,* $\widehat{\mathcal{J}}(\mathbf{z}, \xi) > 0$*, and*

$$\mathbb{E}_{\xi \sim \rho}\Big[\widehat{\mathcal{J}}(\mathbf{z}, \xi)\Big] = |\det D\mathbf{f}(\mathbf{z})| \quad \textit{for } q\textit{-almost every } \mathbf{z}. \tag{13}$$

If $0 < Z_B = \int p_B(\mathbf{x})\, d\mathbf{x} < \infty$*, then the* $\mathbf{z}$*-marginal of the normalized target* $\widetilde{\pi}_1$ *is*

$$\widetilde{\pi}_1(\mathbf{z}) = \frac{p_B(\mathbf{f}(\mathbf{z}))\, |\det D\mathbf{f}(\mathbf{z})|}{Z_B}, \tag{14}$$

and its pushforward through $\mathbf{f}$ *is* $p_B(\mathbf{x})/Z_B$.

Proof. At $\beta = 1$, Eqs. (11)–(12) give the unnormalized joint density $\rho(\xi) p_B(\mathbf{f}(\mathbf{z})) \widehat{\mathcal{J}}(\mathbf{z}, \xi)$. Integrating over ξ and applying Eq. (13) yields $p_B(\mathbf{f}(\mathbf{z}))|\det D\mathbf{f}(\mathbf{z})|$. Its integral over $\mathbf{z}$ equals Z_B by change of variables, and the same change of variables proves the pushforward statement. □

The proposition characterizes the ideal extended target; the finite-M particle population provides an SMC approximation to it. Each particle stores the complete pair $(\mathbf{z}, \xi)$. Initialization draws $(\mathbf{z}, \xi) \sim q\rho$, resampling copies the complete pair, and probe variables are updated through a mutation kernel that leaves $\widetilde{\pi}_{\beta_n}$ invariant. Independent regeneration of the entire probe collection after each incremental-weight update would change the algorithm. Algorithm 1 omits this auxiliary-state bookkeeping for readability.

For a particle carried from level $n-1$, the incremental log weight is

$$\Delta \log \omega_n = (\beta_n - \beta_{n-1}) \log R(\mathbf{z}, \xi) = -(\beta_n - \beta_{n-1}) W(\mathbf{z}, \xi), \tag{15}$$

up to constants common to all particles, where $W = u_X(\mathbf{f}(\mathbf{z})) - u_Z(\mathbf{z}) - \Delta S(\mathbf{z}, \xi)$. The implementation stores Eq. (15) in the log domain for numerical stability. Determinant-level unbiasedness applies to the positive factor $\exp(\Delta S)$ under Eq. (13); Appendices H.3 and B.3 characterize the finite-difference and solver-inversion approximations used in practice.

Normalization, resampling, and mutation. At each level, we add Eq. (15) to the accumulated log weights and normalize them using a log-sum-exp operation, producing $\bar{\omega}_n^{(i)}$. We then compute

$$\mathrm{ESS}_n = \left(\sum_{i=1}^{M} \big(\bar{\omega}_n^{(i)}\big)^2 \right)^{-1}. \tag{16}$$

The implementation uses $\tau = 0.95$: resampling is triggered when $\mathrm{ESS}_n < 0.95M$, with an additional near-endpoint resampling in the reported runs. GMM experiments use systematic resampling, whereas Chignolin uses KL reshuffling [27]; the weights are reset to $1/M$ after either operation, and final observables use the weighted or resampled population specified with each result.

The annealing values are generated identically for both systems. For $j = 0, \ldots, N-1$, the code evaluates

$$\beta_j = \frac{\sigma(-2 + 3j/(N-1)) - \sigma(-2)}{\sigma(1) - \sigma(-2)}, \qquad \sigma(a) = \frac{1}{1+e^{-a}}, \tag{17}$$

which places more intermediate distributions near the ends than a linear schedule.

Mutation uses a coordinate-refresh Metropolis kernel. Let $s = (\mathbf{z}, \xi)$ and let $Q_0(s, s')$ denote the joint coordinate-refresh proposal density. For a general proposal, the Metropolis–Hastings acceptance probability targeting Eq. (12) is

$$\alpha_\beta(s, s') = 1 \wedge \frac{q(\mathbf{z}')\rho(\xi')R(\mathbf{z}', \xi')^\beta Q_0(s', s)}{q(\mathbf{z})\rho(\xi)R(\mathbf{z}, \xi)^\beta Q_0(s, s')}. \tag{18}$$

When Q_0 is reversible with respect to the base law $q\rho$, the base and proposal factors cancel, giving $1 \wedge \exp\{\beta(\log R' - \log R)\}$. The reported coordinate-refresh implementation satisfies this condition by drawing refreshed latent coordinates from their factorized Gaussian base marginals and raw auxiliary coordinates from $\mathcal{N}(0,1)$. The latent refresh standard deviations are $t_{\max} = 1.0$ for GMM and 0.75 for Chignolin; the auxiliary variables have unit standard deviation. At level j, the kernel refreshes random subsets of $K_x(j)$ latent coordinates and $K_\epsilon(j)$ raw auxiliary coordinates. With $b_j = (j/(N-1))^2$, the integer-valued schedules are

$$K_s(j) = \left\lfloor K_{s,\min} b_j^{c_s} + K_{s,\max}\left(1 - b_j^{c_s}\right) \right\rfloor, \qquad s \in \{x, \epsilon\}. \tag{19}$$

Thus the number of refreshed coordinates decreases over annealing. Equation (18) satisfies detailed balance for $\widetilde{\pi}_{\beta_j}$, and base reversibility yields the simplified acceptance rule used in the experiments. All schedules, proposals, and acceptance rules are held fixed across Jacobian-estimator variants within a system.

Diagnostics and reported configurations. Importance ESS and weight-variance diagnostics are computed from the normalized weights immediately before resampling. After resampling, the uniform weights make importance ESS equal to the particle count, so particle ancestry is used to assess diversity: lineage (preservation) ESS and the number or fraction of uniquely preserved ancestors quantify genealogical degeneracy. Final molecular observables are evaluated on the resulting weighted or resampled particle population, as specified with each result. Table 5 lists the matched configurations.

Table 5: Matched SMC configurations used in the high-dimensional experiments. Here $(K_{x,\max} \to K_{x,\min}; c_x)$ and $(K_{\epsilon,\max} \to K_{\epsilon,\min}; c_\epsilon)$ parameterize Eq. (19).

System	M	N	L	τ	Resampling	K_x schedule	K_ϵ schedule
1000D GMM	1,000	2,000	10	.95	systematic	$100 \to 2; .05$	$100 \to 2; .05$
Chignolin	2,000	600	2	.95	KL reshuffling	$200 \to 3; .01$	$200 \to 3; .05$

B Proof of Flow Perturbation++

In this appendix we provide a complete derivation of the Flow Perturbation++ (FP++) estimator under exact inverse step maps, followed by the corresponding scope statement for numerically implemented reverse maps. Assume that every step map $\mathbf{f}_i$ is a C^1 diffeomorphism with a nonsingular Jacobian, that the normalized prior density satisfies $q(\mathbf{z}) > 0$ almost everywhere, and that the observable is integrable under the normalized target density, $\mathbb{E}_{\mathbf{x}\sim p}|O(\mathbf{x})| < \infty$. Consider a flow map $\mathbf{f} : \mathbb{R}^D \to \mathbb{R}^D$ obtained by composing T numerical ODE integration steps:

$$\mathbf{f} = \mathbf{f}_1 \circ \mathbf{f}_2 \circ \cdots \circ \mathbf{f}_T.$$

For an initial state $\mathbf{z} \sim q(\mathbf{z})$ we define a trajectory

$$\mathbf{x}_1 = \mathbf{x}, \quad \mathbf{x}_i = \mathbf{f}_i(\mathbf{x}_{i+1}), \quad \mathbf{x}_{T+1} = \mathbf{z}.$$

Let $\hat{\epsilon}_1, \ldots, \hat{\epsilon}_T$ be i.i.d. random vectors sampled uniformly from the unit sphere S^{D-1}. Define the FP++ action

$$\Delta S(\mathbf{z}, \hat{\epsilon}_1, \ldots, \hat{\epsilon}_T) = -D \sum_{i=1}^{T} \log \left\| \frac{\partial \mathbf{f}_i^{-1}}{\partial \mathbf{x}_i} \hat{\epsilon}_i \right\|.$$

Under these assumptions, we show that

$$\mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[O(\mathbf{x})] = \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}),\, \hat{\epsilon}_{1:T}} \left[O(\mathbf{f}(\mathbf{z})) \frac{p(\mathbf{f}(\mathbf{z}))}{q(\mathbf{z})} \exp\big(\Delta S(\mathbf{z}, \hat{\epsilon}_{1:T})\big) \right]. \tag{20}$$

This identity holds for every test function O, hence it suffices to show that

$$\mathbb{E}_{\hat{\epsilon}_{1:T}} \left[\exp\big(\Delta S(\mathbf{z}, \hat{\epsilon}_{1:T})\big) \right] = \left| \det \frac{\partial \mathbf{f}}{\partial \mathbf{z}} \right|. \tag{21}$$

B.1 Reduction to a single-step identity

Expanding the FP++ weight gives

$$\exp(\Delta S) = \prod_{i=1}^{T} \left\| \frac{\partial \mathbf{f}_i^{-1}}{\partial \mathbf{x}_i} \hat{\epsilon}_i \right\|^{-D}.$$

Because the $\hat{\epsilon}_i$ are independent,

$$\mathbb{E}_{\hat{\epsilon}_{1:T}}[\exp(\Delta S)] = \prod_{i=1}^{T} \mathbb{E}_{\hat{\epsilon}_i} \left[\left\| \frac{\partial \mathbf{f}_i^{-1}}{\partial \mathbf{x}_i} \hat{\epsilon}_i \right\|^{-D} \right].$$

Thus, the main statement reduces to proving the following single-step identity:

$$\mathbb{E}_{\hat{\epsilon}\sim S^{D-1}} \left[\|A\hat{\epsilon}\|^{-D} \right] = |\det(A)|^{-1}, \qquad A \in \mathbb{R}^{D\times D} \text{ invertible.} \tag{22}$$

B.2 Proof of the single-step identity

Let $A = U\Sigma V^\top$ be the SVD of A, where U and V are orthogonal matrices and $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_D)$ contains the singular values of A. Orthogonal matrices preserve the uniform measure on the unit sphere, therefore $V^\top \hat{\epsilon}$ is uniformly distributed on S^{D-1} as well. Using the rotational invariance of the Euclidean norm:

$$\|A\hat{\epsilon}\| = \|U\Sigma V^\top \hat{\epsilon}\| = \|\Sigma v\|, \qquad v = V^\top \hat{\epsilon}.$$

Thus,

$$\mathbb{E}_{\hat{\epsilon}}[\|A\hat{\epsilon}\|^{-D}] = \mathbb{E}_{v\sim S^{D-1}} \left[\left(\sum_{j=1}^{D} \sigma_j^2 v_j^2 \right)^{-D/2} \right].$$

A classical result from integral geometry (obtainable via a change to ellipsoidal coordinates or by comparing Gaussian integrals in two ways) states that

$$\mathbb{E}_{v\sim S^{D-1}} \left[\left(\sum_{j=1}^{D} \sigma_j^2 v_j^2 \right)^{-D/2} \right] = \frac{1}{\prod_{j=1}^{D} \sigma_j}. \tag{23}$$

Since $\prod_{j=1}^{D} \sigma_j = |\det(A)|$, equation (22) follows immediately.

B.3 Completion of the FP++ proof

Applying (22) to each integration step with $A_i = \frac{\partial \mathbf{f}_i^{-1}}{\partial \mathbf{x}_i}$ yields

$$\mathbb{E}_{\hat{\epsilon}_{1:T}}[\exp(\Delta S)] = \prod_{i=1}^{T} \left| \det \left(\frac{\partial \mathbf{f}_i^{-1}}{\partial \mathbf{x}_i} \right) \right|^{-1} = \prod_{i=1}^{T} \left| \det \left(\frac{\partial \mathbf{f}_i}{\partial \mathbf{x}_{i+1}} \right) \right|.$$

By the chain rule for Jacobians,

$$\prod_{i=1}^{T}\left|\det\left(\frac{\partial \mathbf{f}_i}{\partial \mathbf{x}_{i+1}}\right)\right| = \left|\det\left(\frac{\partial \mathbf{f}}{\partial \mathbf{z}}\right)\right|.$$

Substituting into (20) yields the FP++ importance sampling identity, completing the proof. □

The proof assumes exact inverse-step directional derivatives. Finite differences introduce a numerical approximation, so exact unbiasedness does not directly extend to the implementation. Appendix H.3 assesses its agreement with the JVP reference for the implemented reverse map.

C Flow Perturbation++ variance reduction analysis

In this section, we analyze the log-domain variance of the Flow Perturbation++ (FP++) estimator relative to the full-flow Flow Perturbation (FP) estimator, conditional on a fixed flow trajectory. The variance comparison depends on the one-step marginal distributions and cross-step covariances.

Recall that in FP++, we use independent random perturbations for each flow step, whereas in FP, the same initial perturbation is propagated through all flow steps. For simplicity, denote the contribution of the i-th step as

$$X_i = -D\log\left\|\frac{\partial \mathbf{f}_i^{-1}}{\partial \mathbf{x}_i}\hat{\epsilon}_i\right\|, \tag{24}$$

where $\hat{\epsilon}_i$ is the independent unit random perturbation at step i in FP++. In this appendix, write $J_i = \partial \mathbf{f}_i^{-1}/\partial \mathbf{x}_i$ for the inverse-step Jacobian. Starting from $\hat{\epsilon}_0^{\mathrm{FP}} = \hat{\epsilon} \sim \mathrm{Unif}(S^{D-1})$, the direction in full-flow FP is propagated as

$$\hat{\epsilon}_i^{\mathrm{FP}} = \frac{J_i\hat{\epsilon}_{i-1}^{\mathrm{FP}}}{\|J_i\hat{\epsilon}_{i-1}^{\mathrm{FP}}\|}, \qquad Y_i = -D\log\|J_i\hat{\epsilon}_{i-1}^{\mathrm{FP}}\|. \tag{25}$$

The total contributions are

$$\Delta S_{\mathrm{FP++}} = \sum_i X_i, \qquad \Delta S_{\mathrm{FP}} = -D\log\|J_T\cdots J_1\hat{\epsilon}\| = \sum_i Y_i. \tag{26}$$

Thus full-flow FP does not use an unchanged direction at every step.

C.1 Single-step variance

When each flow step acts on random unit vectors with identical distributions in FP and FP++, the single-step variances are equal:

$$\mathrm{Var}[X_i] = \mathrm{Var}[Y_i]. \tag{27}$$

C.2 Covariance structure

The total variance for FP is

$$\mathrm{Var}\left[\sum_i Y_i\right] = \sum_i \mathrm{Var}[Y_i] + 2\sum_{i<j}\mathrm{Cov}[Y_i, Y_j]. \tag{28}$$

The covariance is

$$\mathrm{Cov}[Y_i, Y_j] = \mathbb{E}[Y_iY_j] - \mathbb{E}[Y_i]\mathbb{E}[Y_j]. \tag{29}$$

Define

$$g_i(\hat{\epsilon}) = \log\|J_i\hat{\epsilon}\|, \quad g_j(\hat{\epsilon}) = \log\|J_j\hat{\epsilon}\|. \tag{30}$$

For a unit vector $\hat{\epsilon}$, we can write

$$g_i(\hat{\epsilon}) = \frac{1}{2}\log(\hat{\epsilon}^\top A_i\hat{\epsilon}), \quad g_j(\hat{\epsilon}) = \frac{1}{2}\log(\hat{\epsilon}^\top A_j\hat{\epsilon}), \tag{31}$$

where $A_i = J_i^\top J_i$ and $A_j = J_j^\top J_j$ are symmetric positive definite matrices. Then, for the propagated directions in Eq. (25),

$$\mathrm{Cov}[Y_i, Y_j] = D^2\,\mathrm{Cov}\left[g_i(\hat{\epsilon}_{i-1}^{\mathrm{FP}}), g_j(\hat{\epsilon}_{j-1}^{\mathrm{FP}})\right]. \tag{32}$$

This covariance need not be non-negative for general step Jacobians.

Local covariance analysis. For nearby integration steps, aligned anisotropic stretching can give positive aggregate covariance. Consider small steps $J_i = I + h_i B_i + O(h^2)$, where $h_i = h\alpha_i$ with fixed α_i, dimension, and number of steps. Let $S_i = (B_i + B_i^\top)/2$ and $\widetilde{S}_i = S_i - \mathrm{tr}(S_i)I/D$.

Lemma C.1 (Local alignment condition)**.** *With uniformly bounded remainders,*

$$\sum_{i<j} \mathrm{Cov}[Y_i, Y_j] = \frac{2D}{D+2} \sum_{i<j} h_i h_j \langle \widetilde{S}_i, \widetilde{S}_j \rangle_F + O(h^3). \tag{33}$$

If $\kappa = \sum_{i<j} \alpha_i \alpha_j \langle \widetilde{S}_i, \widetilde{S}_j \rangle_F > 0$*, then* $\sum_{i<j} \mathrm{Cov}[Y_i, Y_j] > 0$ *for all sufficiently small positive* h*. The sufficient condition applies in this small-step regime.*

Proof. For a uniform initial direction $\hat{\epsilon}$, the propagated direction is $\hat{\epsilon}_{i-1}^{\mathrm{FP}} = \hat{\epsilon} + O(h)$, so $Y_i = -Dh_i \hat{\epsilon}^\top S_i \hat{\epsilon} + O(h^2)$. The spherical covariance identity is

$$\mathrm{Cov}(\hat{\epsilon}^\top S_i \hat{\epsilon}, \hat{\epsilon}^\top S_j \hat{\epsilon}) = \frac{2\langle \widetilde{S}_i, \widetilde{S}_j \rangle_F}{D(D+2)}.$$

Multiplying by $D^2 h_i h_j$ and summing gives Eq. (33); the bounded remainder contributes $O(h^3)$. When $\kappa > 0$, the sum is $ch^2 + R(h)$ with $c = 2D\kappa/(D+2) > 0$ and $|R(h)| \le Kh^3$ for some finite $K \ge 0$. Choosing $h > 0$ small enough that $Kh < c/2$ yields $\sum_{i<j} \mathrm{Cov}[Y_i, Y_j] \ge ch^2/2 > 0$. □

C.3 Variance reduction in FP++

For FP++, using independent perturbations, the covariance terms vanish: under the matched one-step marginal condition in Section C.1 and the positive aggregate covariance condition established by Lemma C.1,

$$\mathrm{Var}\Big[\sum_i X_i\Big] = \sum_i \mathrm{Var}[X_i] = \sum_i \mathrm{Var}[Y_i] < \mathrm{Var}\Big[\sum_i Y_i\Big]. \tag{34}$$

Thus, under these conditions, the total log-domain variance of the FP++ estimator is strictly smaller than that of FP. Intuitively, breaking the dependence between steps prevents positive aggregate correlation between contributions from inflating the overall variance. With matched one-step marginals, zero aggregate covariance gives equal variances; the ordering for general Jacobians depends on both the marginals and the covariances.

C.4 Summary

- The key difference is that FP++ uses independent random directions per step, whereas FP propagates its initial direction through the inverse flow.
- The single-step variances are equal only when the marginal distributions match.
- Removing cross-step dependence gives variance reduction under the stated marginal and covariance conditions; nearby-step alignment supplies a sufficient local condition.
- Lower log-domain variance alone does not imply lower determinant or importance-weight variance. Appendix H.5 reports direct determinant/weight diagnostics, while Appendix H.1 reports log-domain measurements.

D Gaussian mixture model benchmark

In this work, we consider a Gaussian Mixture Model (GMM) to generate a multimodal test distribution. An n-dimensional GMM with k components represents the probability density as

$$p(\mathbf{x}) = \sum_{j=1}^{k} \pi_j \, \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_j, \Sigma_j), \tag{35}$$

where π_j is the mixture weight for the j-th Gaussian, and $\mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_j, \Sigma_j)$ denotes the corresponding multivariate normal distribution.

For our experiments, we adopt a two-component GMM ($k = 2$) with the following setup:

- **Component Means ($\boldsymbol{\mu}_j$)**: The first Gaussian is centered at $(-2, 0, \ldots, 0)$, and the second at $(2, 0, \ldots, 0)$, with the displacement applied only along the first dimension and all remaining dimensions set to zero.

- **Covariance Matrices (Σ_j)**: The two components have distinct covariance structures: - Component 1 ($j = 1$) uses a diagonal covariance matrix with positive entries drawn as $0.01 + |\mathcal{N}(0, 0.25)|$, ensuring positive definiteness. - Component 2 ($j = 2$) employs a full covariance matrix: first, a diagonal matrix is sampled as above, then a random rotation (obtained via the QR decomposition of a random matrix) is applied to introduce correlations between dimensions.

- **Mixing Weights (π_j)**: We set $\pi_0 = 0.25$ and $\pi_1 = 0.75$, creating an asymmetric mixture with the second component more heavily weighted.

The density can equivalently be expressed as a Boltzmann distribution,

$$p(\mathbf{x}) = \exp(-E(\mathbf{x})), \tag{36}$$

where $E(\mathbf{x})$ denotes the energy of configuration $\mathbf{x}$.

This particular GMM presents a challenging scenario for sampling, with two well-separated modes and heterogeneous covariance structures, making it a suitable benchmark for testing and validating advanced Boltzmann sampling techniques.

E Brute-force and stochastic estimation of CNF Jacobian traces

Continuous Normalizing Flows (CNFs) generalize standard Normalizing Flows by defining the mapping $\mathbf{f} : \mathbf{z} \to \mathbf{x}$ through an ordinary differential equation (ODE):

$$\frac{d\mathbf{x}_t}{dt} = \mathbf{v}(\mathbf{x}_t, t), \quad \mathbf{x}_0 = \mathbf{z}, \tag{37}$$

where $\mathbf{v}(\mathbf{x}, t)$ denotes the velocity field governing the transformation. The log-determinant of the flow Jacobian can then be expressed as an integral over the divergence of $\mathbf{v}$:

$$\log\Big|\det \frac{\partial \mathbf{f}}{\partial \mathbf{z}}\Big| = \int_0^1 \nabla \cdot \mathbf{v}(\mathbf{x}_t, t)\, dt, \tag{38}$$

with the divergence given by the trace of the Jacobian $\partial \mathbf{v}/\partial \mathbf{x}$.

We outline two standard approaches for computing this Jacobian trace: a direct (brute-force) method and a stochastic method using the Hutchinson estimator.

E.1 Direct Jacobian construction via automatic differentiation

Let $\mathbf{v} : \mathbb{R}^D \to \mathbb{R}^D$ be differentiable. A full Jacobian $\partial \mathbf{v}/\partial \mathbf{x}$ can be obtained using automatic differentiation frameworks (e.g., PyTorch):

1. For each output component $v_i(\mathbf{x})$, define the unit vector $\mathbf{e}_i \in \mathbb{R}^D$ with a 1 in the i-th position and zeros elsewhere, such that
$$v_i(\mathbf{x}) = \mathbf{e}_i^\top \mathbf{v}(\mathbf{x}).$$
2. Compute the gradient $\nabla_{\mathbf{x}} v_i(\mathbf{x})$ via backpropagation. This yields the i-th row of the Jacobian:
$$\nabla_{\mathbf{x}} v_i(\mathbf{x}) = \Big[\frac{\partial v_i}{\partial x_1}, \frac{\partial v_i}{\partial x_2}, \ldots, \frac{\partial v_i}{\partial x_D}\Big].$$
3. Stack all rows to assemble the full Jacobian:
$$\frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \begin{bmatrix} \nabla_{\mathbf{x}} v_1(\mathbf{x}) \\ \nabla_{\mathbf{x}} v_2(\mathbf{x}) \\ \vdots \\ \nabla_{\mathbf{x}} v_D(\mathbf{x}) \end{bmatrix}.$$

In practice, PyTorch provides `jacrev` to efficiently compute this Jacobian.

E.2 Hutchinson estimator for trace approximation

The Hutchinson estimator offers a stochastic alternative for estimating $\mathrm{Tr}(\partial \mathbf{v}/\partial \mathbf{x})$ without explicitly forming the full Jacobian:

1. Sample a random vector $\mathbf{u} \in \mathbb{R}^D$ from a standard normal or Rademacher distribution.
2. Compute the gradient of the scalar $\mathbf{u}^\top \mathbf{v}(\mathbf{x})$ w.r.t. $\mathbf{x}$:
$$\mathbf{g} = \nabla_{\mathbf{x}}(\mathbf{u}^\top \mathbf{v}(\mathbf{x})) = \frac{\partial(\mathbf{u}^\top \mathbf{v})}{\partial \mathbf{x}}.$$
3. Form the quadratic form $\mathbf{u}^\top \mathbf{g} = \mathbf{u}^\top (\partial \mathbf{v}/\partial \mathbf{x})\, \mathbf{u}$, which is an unbiased estimator of the trace.
4. Repeat for N independent random vectors $\mathbf{u}_i$ and average:
$$\mathrm{Tr}\Big(\frac{\partial \mathbf{v}}{\partial \mathbf{x}}\Big) \approx \frac{1}{N} \sum_{i=1}^{N} \mathbf{u}_i^\top \frac{\partial \mathbf{v}}{\partial \mathbf{x}}\, \mathbf{u}_i.$$

Each sample requires one backpropagation pass. Averaging over multiple $\mathbf{u}_i$ reduces estimator variance, providing an efficient method for high-dimensional CNFs.

F Computational cost of JVP/VJP-based estimators

F.1 FP and FP++ implementations: FD, JVP, and VJP

The inverse–Jacobian–vector product appearing in Eq. 7, $J_k^{-1}\epsilon_k = (\partial \mathbf{f}_k^{-1}/\partial \mathbf{x}_k)\, \epsilon_k$, can be implemented in three ways. In practice, the reverse integration approximates $\mathbf{f}_k^{-1}$; a JVP of this implemented map is exact for that map, not necessarily for the true inverse (Appendix B.3).

Finite-difference (FD) approximation. The product is approximated by perturbing the inverse flow:
$$J_k^{-1}\epsilon_k \approx \frac{\mathbf{f}_k^{-1}(\mathbf{x}_k + \delta\epsilon_k) - \mathbf{f}_k^{-1}(\mathbf{x}_k - \delta\epsilon_k)}{2\delta}.$$

Forward-mode Jacobian-vector product (JVP). The JVP computes the exact directional derivative along ϵ_k:
$$\mathrm{JVP}(\mathbf{f}_k^{-1}, \mathbf{x}_k, \epsilon_k) = \frac{d}{d\eta} \mathbf{f}_k^{-1}(\mathbf{x}_k + \eta\epsilon_k)\Big|_{\eta=0} = J_k^{-1}\epsilon_k.$$

Reverse-mode vector-Jacobian product (VJP). Given a cotangent ϵ_k at the output, the VJP computes
$$\mathrm{VJP}(\mathbf{f}_k^{-1}, \mathbf{x}_k, \epsilon_k) = \epsilon_k^\top J_k^{-1}.$$
For the same probe, the VJP and JVP norms need not be equal. Their squared norms involve the inverse Jacobian and its transpose in opposite orders; these matrices have the same eigenvalues. Uniform spherical probes therefore give identical norm distributions, rather than pointwise equality.

Delta entropy computation. For all three implementations, the contribution to the flow-based entropy change at step k is
$$\Delta S_k = -\frac{D}{2} \log \frac{\|J_k^{-1}\epsilon_k\|^2}{\|\epsilon_k\|^2},$$

F.2 FD-based implementations in the main text

In the main text, FP and FP++ are implemented using *finite-difference* (FD) estimators. Here, Jacobian information is approximated by evaluating the inverse flow at perturbed inputs. This requires:

- 1 ODE pass for trajectory generation, and
- 2 additional ODE passes for entropy estimation,

yielding a total of 3 ODE passes per SMC step.

F.3 JVP/VJP-based implementations

In this appendix and in Table 6, we report runtimes for implementations that use explicit vector–Jacobian (VJP) or Jacobian–vector (JVP) products. In this regime, entropy-related terms are computed via automatic differentiation without perturbing the inverse flow. As a result, FP and FP++ require:

- 1 ODE pass for trajectory propagation, and
- 1 ODE pass for the associated JVP/VJP computation,

yielding a total of 2 ODE passes per SMC step. In the reported comparison, the G2/R2 and G10/R10 Hutchinson variants use two and ten stochastic probes, respectively, resulting in more ODE passes and longer wall-clock times.

F.4 Wall-clock times and interpretation

Table 6 summarizes both the theoretical (ODE counts) and empirical (wall-clock) computational cost for different Jacobian estimators under the JVP/VJP regime. The columns **VJP Time** and **JVP Time** report measured wall-clock times for explicit vector–Jacobian and Jacobian–vector implementations, respectively. The results indicate that, for JVP/VJP-based implementations, the dominant computational cost arises from the number of ODE solver evaluations.

Table 6: Computational cost of different Jacobian estimators. Cost (# ODE passes) counts the number of ODE solver evaluations per SMC step, including trajectory propagation and entropy computation. VJP Time and JVP Time report the total wall-clock time incurred by implementations based on explicit vector–Jacobian or Jacobian–vector products, respectively.

Method	Cost (# ODE passes)	VJP Time	JVP Time
Hutchinson G2 / R2	3	30.4 h	48.9 h
Hutchinson G10 / R10	11	100 h	176 h
FP (single-step)	2	23.8 h	30.8 h
FP++ (multi-step)	2	23.8 h	30.8 h

F.5 Step count, intermediate states, and memory

For multiple probes at a fixed input, one primal linearization can be constructed and its linear map reused for all probe directions. This requires one primal evaluation and one linear-map application per probe, rather than two perturbed inverse evaluations per finite-difference probe. Retaining the reusable linearization introduces a memory trade-off. The timings above cover the evaluated implementations; the reuse strategy requires a separate runtime measurement.

The ODE-equivalent counts in Tables 1 and 6 treat one complete trajectory solve as one pass. They therefore hide solver-dependent constants: wall-clock cost grows with the number of integration steps and stages per step, and adaptive solvers add control-flow and rejected-step overhead. State management also matters. A direct per-step implementation may retain $T + 1$ trajectory states for each particle, requiring $O(MTD)$ state storage for M particles in D dimensions. Streaming reverse evaluation or checkpointing can reduce resident state storage toward $O(MD)$, or an intermediate checkpoint-dependent cost, by recomputing portions of the trajectory. JVP/VJP implementations may additionally retain or reconstruct an automatic-differentiation graph, whereas finite differences use forward evaluations and avoid that graph at the cost of two perturbed solves per probe. Table 13 provides a runtime and memory comparison for alanine dipeptide under the evaluated solver settings. A systematic comparison across solver families, tolerances, and step counts remains outside the scope of these measurements.

G Molecular dynamics simulations of Chignolin mutant

We performed molecular dynamics (MD) simulations to investigate the behavior of the Chignolin Mutant. A total of six independent trajectories were generated, each consisting of 10^7 integration steps with a time step of 1 fs. All simulations were carried out at a temperature of 300 K, employing

the CHARMM22 force field together with the implicit OBC2 solvent model. These conditions are consistent with those used for producing the training dataset.

Across all runs, the system remained confined within one of its metastable conformational states. Figure 6 illustrates two representative trajectories projected onto the chosen reaction coordinate (RC) plane.

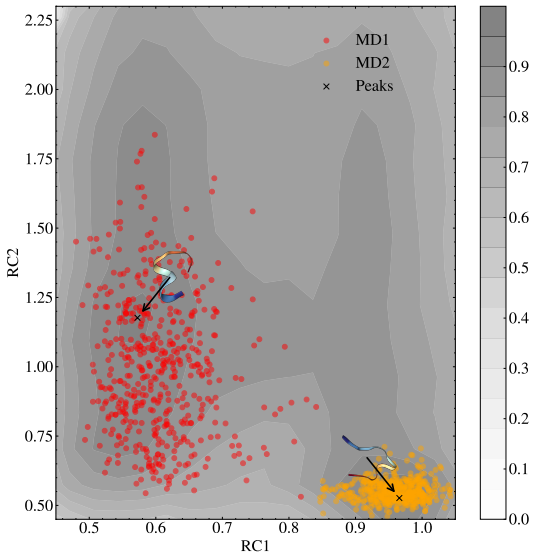


Figure 6: Representative MD trajectories of Chignolin Mutant visualized on the reaction coordinate plane.

H Additional experimental results

This section provides supplementary experimental results that further evaluate the accuracy, stability, and convergence behavior of the proposed estimators.

H.1 Variance of the log-domain estimator

We evaluate the variance of the logged Jacobian-correction estimates across methods and dimensionalities using 1000 samples. These quantities provide the stable numerical representation used in computation, whereas determinant-level unbiasedness pertains to their exponentiated positive factors. Table 7 summarizes the results.

Setup. We consider Gaussian mixture models (GMMs) with dimensions ranging from 10D to 1000D. BF Jacobian denotes the exact computation and serves as ground truth. Methods with prefix FP use finite-difference estimators (FP/FP++), where the number indicates the number of probes per step (e.g., FP1 uses one probe). Methods denoted as Gk and Rk correspond to Hutchinson estimators using Gaussian and Rademacher probes, respectively.

Results. We observe that stochastic estimators such as Hutchinson exhibit increasing variance as dimensionality grows, especially under small probe budgets. Increasing the number of probes generally reduces variance but incurs higher computational cost.

In contrast, FP++ consistently achieves log-domain variance close to the BF Jacobian across all dimensions, even with a single probe ($N = 1$). Notably, FP++ maintains stable log-domain variance as dimensionality increases, whereas the single-step FP estimator exhibits significantly higher variance, particularly in high dimensions.

These results demonstrate that the multi-step aggregation in FP++ achieves significantly lower log-domain variance than FP and low-probe Hutchinson-based estimators in these experiments, while

Table 7: Variance of logged Jacobian-correction estimates (1000 samples) across methods and dimensionalities.

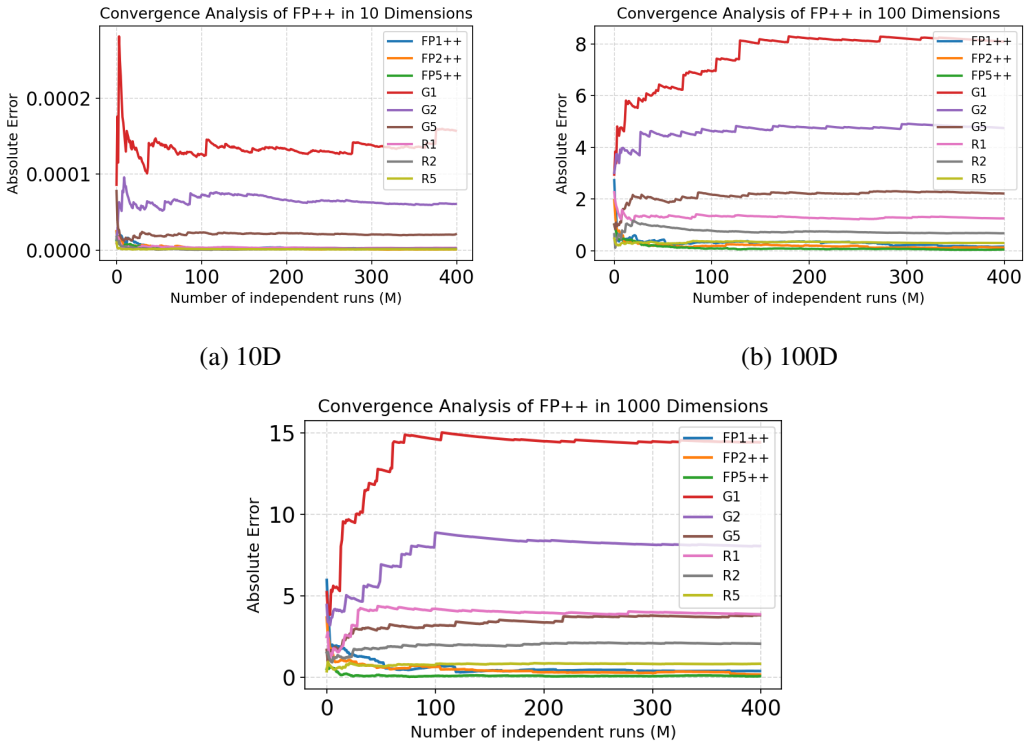
Method	GMM 10D	GMM 100D	GMM 1000D
BF Jacobian	0.9265	1.5931	11.0544
FP1	1.6886	10.2365	37.6025
FP2	1.4847	8.0238	35.3451
FP5	1.1487	6.5414	31.2574
FP10	0.9597	5.3718	29.4457
FP1++	0.9456	1.8040	11.4718
FP2++	0.9393	1.6681	11.3622
FP5++	0.9259	1.6174	11.1046
FP10++	0.9255	1.6193	11.1099
G1	1.1992	2.7067	12.8340
G2	1.1018	2.1713	12.0969
G5	0.9930	1.8662	11.5580
G10	0.9503	1.7211	11.2355
R1	0.9394	1.7396	11.3268
R2	0.9378	1.6757	11.1838
R5	0.9266	1.6275	11.1346
R10	0.9281	1.6109	11.0810

reaching a variance level comparable to the brute-force Jacobian, making it a more accurate and robust estimator under the same computational budget. Appendix H.5 separately evaluates determinant and weight variance.

H.2 Convergence analysis of FP++

We study the convergence behavior of FP++ with respect to the number of independent runs M. The estimation error is defined as the average absolute difference between the estimated Jacobian determinant and the ground-truth value, averaged over both independent runs and a fixed set of samples (number = 5).

Results. Figure 7 reports the running-average errors in 10D, 100D, and 1000D Gaussian mixture models. As M grows, the curves settle toward method-dependent plateaus rather than decreasing monotonically in every case. These plateaus separate finite-repetition noise from residual estimator or numerical error relative to the brute-force reference and complement the log-domain variance comparison in Section H.1.



(a) 10D

(b) 100D

(c) 1000D

Figure 7: Running-average error as the number of independent repetitions M increases. Curves stabilize at method-dependent residual errors in 10D, 100D, and 1000D.

H.3 Effect of finite-difference step size

We analyze the sensitivity of FP++ to the finite-difference step size δ and evaluate its numerical stability across different dimensionalities. We additionally compare finite-difference FP++ with a Jacobian-vector-product (JVP) implementation. This comparison isolates finite-difference error relative to the implemented reverse map, while Appendix B.3 characterizes the complementary solver-inversion error.

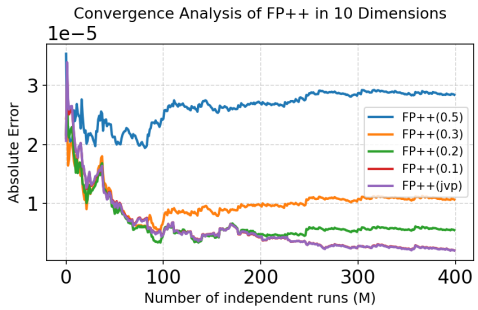
Setup. We evaluate FP++ with different choices of δ, where the notation FP++(δ) denotes the estimator using step size δ. Experiments are conducted on Gaussian mixture models (10D, 100D, 1000D) and a molecular system (Chignolin mutant).

Results. The results are shown in Figure 8. We observe the following:

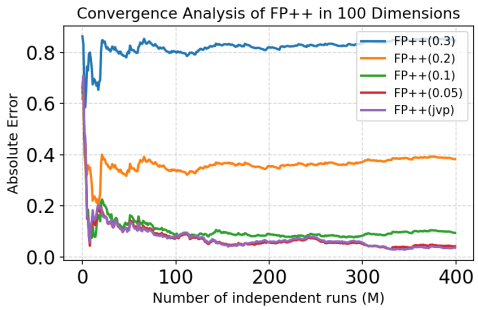
- In low-dimensional settings (10D), finite-difference FP++ closely matches the JVP result even with relatively large step sizes (e.g., $\delta = 0.1$).
- In higher dimensions (100D and 1000D), smaller step sizes (e.g., $\delta = 0.05$) are sufficient to achieve stable and consistent estimates.
- In the more complex Chignolin system, a smaller step size ($\delta = 0.001$) is required to maintain high numerical accuracy and agreement with JVP.

Overall, as δ decreases, the finite-difference FP++ estimate approaches the JVP-based result before entering a roundoff-sensitive regime.

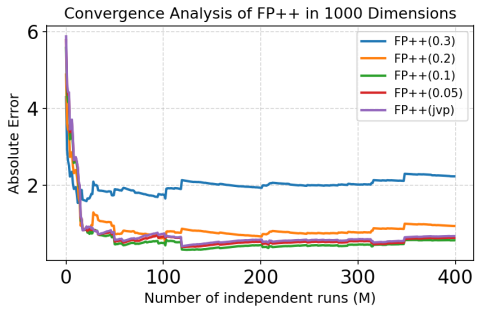
In all main experiments, we adopt $\delta = 0.001$, which lies in the visually stable regime of Figure 8 and agrees closely with the JVP reference at the plotted resolution.



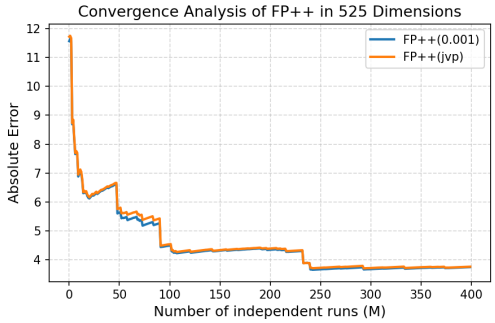
(a) 10D GMM



(b) 100D GMM



(c) 1000D GMM



(d) Chignolin mutant

Figure 8: Effect of finite-difference step size δ on FP++. As δ decreases, finite-difference FP++ converges to the JVP-based result across different dimensionalities and systems.

H.4 Stability on Chignolin system

We evaluate the stability of FP++ on the Chignolin system across 10 independent runs. We report two metrics:

- **Proportion**: the fraction of particles located in the right basin, corresponding to the β-hairpin configuration;
- **# Preserved**: the number of preserved lineages after resampling, measuring particle diversity.

Results. The results are summarized in Table 8. We observe that FP++ exhibits highly consistent performance across runs.

Specifically, the proportion of particles in the target basin remains stable (mean 0.3658 with low standard deviation 0.0146), indicating reliable sampling of the metastable state. At the same time, the number of preserved lineages remains consistently high (mean 1473.8), suggesting that FP++ maintains strong particle diversity after resampling.

These results demonstrate that FP++ achieves both stable sampling behavior and effective diversity preservation in a realistic molecular system.

Table 8: Stability evaluation on the Chignolin system across 10 independent runs.

Method	Metric	1 / 6	2 / 7	3 / 8	4 / 9	5 / 10	Mean $\pm$ Std
FP++	Proportion	0.3640 0.3855	0.3745 0.3775	0.3600 0.3580	0.3455 0.3515	0.3540 0.3875	0.3658 ± 0.0146
FP++	# Preserved	1463 1477	1495 1491	1461 1447	1451 1480	1498 1475	1473.8 ± 18.0

Matched ten-run statistics for all estimators are reported in Table 9. FP++ has the mean closest to the target 0.385 and the smallest basin-fraction standard deviation among the tested estimators. FP occasionally approaches the target but varies more strongly across runs, while all Hutchinson variants underestimate the target on average.

Table 9: Chignolin repeated-run statistics under matched settings. All methods use ten independent runs.

Method	Basin fraction	Range	# Preserved
FP++	0.3658 ± 0.0146	0.3455–0.3875	1473.8 ± 18.0
FP	0.4450 ± 0.0510	0.3320–0.4830	1432.7 ± 28.3
R2	0.2330 ± 0.0168	0.2115–0.2675	1529.3 ± 25.4
G2	0.2136 ± 0.0685	0.1755–0.4065	1315.7 ± 39.7
R10	0.2457 ± 0.0369	0.2100–0.3210	1541.1 ± 29.8
G10	0.2303 ± 0.0394	0.1975–0.3260	1458.4 ± 19.0

Table 10 reports complementary Chignolin diagnostics from representative single runs. Importance ESS is measured before resampling; lineage ESS accounts for common ancestry after resampling. Unique/N is the fraction of distinct surviving lineages. Energy-W2 is measured in k_BT and torsion-W2 in radians.

Table 10: Representative-run Chignolin diagnostics with 2,000 particles.

Method	Basin	Energy-W2	Torsion-W2	Lin. ESS/N	Imp. ESS/N	Unique/N
FP	0.4750	1.6512	3.0853	0.036647	0.001190	0.7140
FP++	0.3640	2.7330	2.7349	0.043632	0.003499	0.7315
R2	0.2255	30.1826	2.7397	0.084090	0.002502	0.7685
R10	0.2240	6.7210	2.6891	0.062073	0.001761	0.7675
G2	0.1945	31.0080	2.8770	0.014943	0.000534	0.6650
G10	0.2220	11.1386	2.7375	0.036490	0.002464	0.7225

H.5 Fixed-input determinant and weight variance

To separate estimator randomness from configuration variation, we use a two-well GMM with an analytically available determinant: $\frac{1}{2}\mathcal{N}(-1.5\mathbf{d}, 0.9^2\mathbf{I}) + \frac{1}{2}\mathcal{N}(1.5\mathbf{d}, 0.9^2\mathbf{I})$, where $\mathbf{d} = (1, \ldots, 1)/\sqrt{D}$. An exact quantile transport is decomposed into eight invertible steps; inversion errors are below 6.7×10^{-15}. We evaluate 16 fixed latent quantiles per dimension and repeat each estimator 5,000 times per input. Here $\widehat{J}/J$ is also the induced importance-weight ratio. Table 11 summarizes the diagnostics. FP1++ has lower determinant variance than FP1 for all 16 fixed inputs in each dimension. The median FP1/FP1++ variance ratios are 19.99×, 19.03×, and 22.82× in 10D, 100D, and 1000D, respectively. These fixed-input measurements show that FP++ reduces determinant and importance-weight variance relative to FP under the evaluated benchmark conditions.

H.6 Feynman–Kac correctors on GMM benchmark

We evaluate Feynman–Kac corrector (FKC)-based methods[49] on a 10D Gaussian Mixture Model (GMM) with an intentionally imbalanced target distribution. The metric *neg ratio* denotes the fraction of samples whose first dimension is less than zero, corresponding to one mode of the mixture. The ground-truth target ratio is 0.25, reflecting the desired 25/75 mixture proportion.

Setup. We consider two settings: (i) annealed SDE sampling with FKC correction, and (ii) reward-tilted sampling with FKC. Additionally, we compare against RNE-based methods. All methods are evaluated under the same proposal distribution.

Results. Figure 9 summarizes the results. We observe that FKC-based[49] and RNE-based [17] methods fail to recover the target mixture imbalance, instead remaining close to the bias of the proposal distribution. In contrast, FP++ is able to accurately recover the correct mode proportion.

Table 11: Fixed-input determinant and importance-weight diagnostics, aggregated over 16 fixed inputs per dimension and 5,000 estimator repetitions per input.

D	Method	Mean	Var.	CV^2	ESS/N
10	FP1	0.98836	15.20617	13.72306	0.44239
10	FP2	0.98621	7.39192	7.32949	0.53670
10	FP10	1.01540	5.91418	5.11221	0.67136
10	FP1++	0.99862	0.15049	0.15189	0.88678
10	FP2++	0.99951	0.07103	0.07135	0.93870
10	FP10++	1.00008	0.01364	0.01364	0.98678
100	FP1	0.96615	90.14904	42.21219	0.34038
100	FP2	0.96248	35.38961	25.97286	0.42131
100	FP10	0.98688	38.01803	31.88503	0.52187
100	FP1++	1.00348	0.24940	0.24687	0.84224
100	FP2++	1.00150	0.12210	0.12154	0.90734
100	FP10++	1.00013	0.02134	0.02131	0.97979
1000	FP1	0.95217	19.12200	21.79667	0.35077
1000	FP2	0.94331	6.79784	9.22970	0.44994
1000	FP10	0.94874	2.52255	3.48656	0.56135
1000	FP1++	0.99910	0.33129	0.33086	0.82973
1000	FP2++	1.00030	0.17903	0.17742	0.89120
1000	FP10++	1.00077	0.02629	0.02609	0.97576

Discussion. This behavior can be explained by the fact that FKC and RNE methods do not explicitly compute absolute density ratios. As a result, their induced distributions remain closely tied to the proposal distribution, rather than correcting for the true target imbalance.

In contrast, FP++ incorporates the Jacobian determinant term, which explicitly accounts for volume changes under the transformation. This enables FP++ to correctly reweight the induced distribution and recover the target mixture proportions independent of the proposal bias.

H.7 1000D analytical Jacobian benchmark

We further validate FP++ on a fully analytical 1000D Gaussian Mixture Model, where both the velocity field and its Jacobian determinant are available in closed form. This setting provides an exact reference and removes learned-model error from the Jacobian-estimation comparison.

Setup. We randomly sample initial points and evaluate the log-Jacobian determinant along the continuous trajectory $t \in [0, 1]$. At each time step, results are averaged over multiple samples to reduce statistical noise.

Results. Figure 10 shows the evolution of the log-Jacobian determinant over time. We observe that FP++ closely matches the analytical ground truth throughout the entire trajectory, maintaining high accuracy even in 1000-dimensional space.

Conclusion. This experiment confirms that FP++ provides numerically consistent and accurate Jacobian determinant estimation under fully analytical conditions.

H.8 FP++ Jacobian estimation with TBG pretrained models

We further evaluate FP++ on a realistic molecular system using a pretrained Transferable Boltzmann Generators (TBG) model [24]. We consider the alanine dipeptide system as a benchmark for thermodynamic consistency and free energy estimation.

TBG is trained via equivariant continuous normalizing flows and can generate molecular configurations across different temperatures. However, its induced distribution generally deviates from the true Boltzmann distribution, leading to biased thermodynamic estimates.

FP++ addresses this issue by providing a low-cost determinant estimator along the flow trajectory. Its ideal exact-inverse construction is unbiased at the determinant level, while finite-difference and

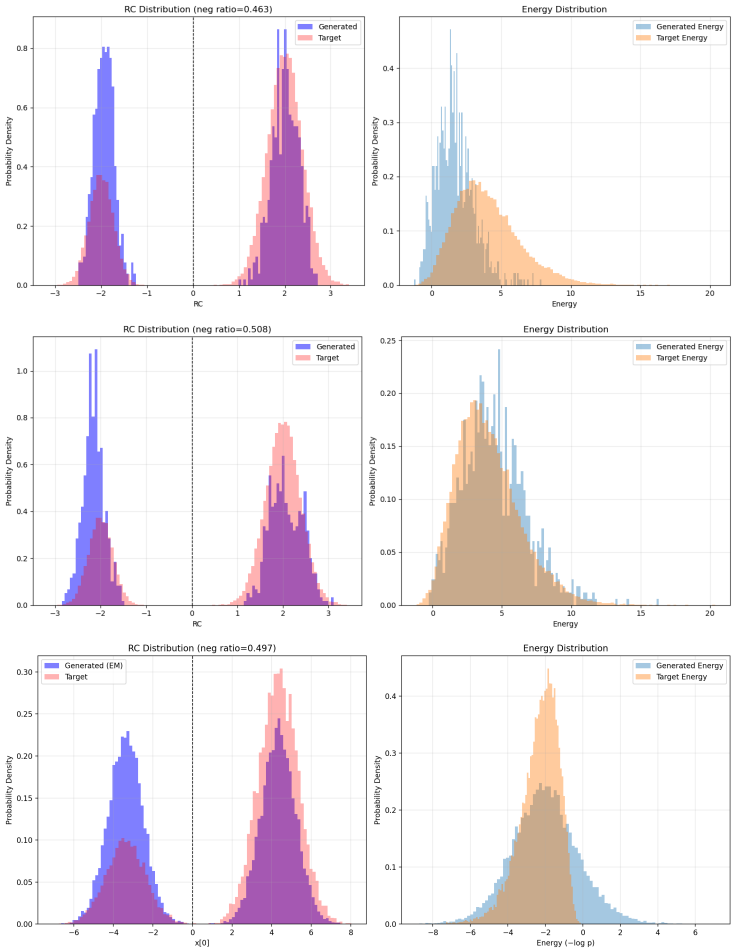


Figure 9: FKC and RNE results on the 10D GMM benchmark. The target neg ratio is 0.25 (true mixture proportion). FKC and RNE methods fail to correct the inherited bias of the proposal distribution, while FP++ successfully recovers the correct mixture proportions.

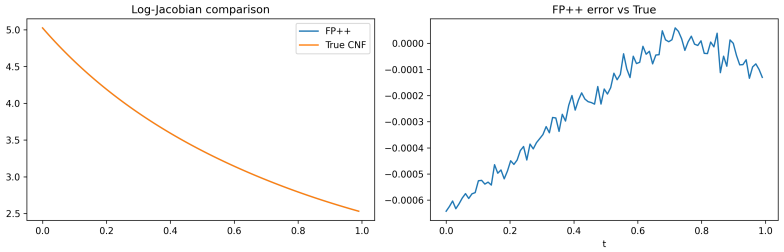


Figure 10: Jacobian determinant accuracy under a fully analytical 1000D Gaussian Mixture Model. FP++ closely matches the ground-truth log-Jacobian determinant over the entire trajectory $t \in [0, 1]$.

numerical-inversion effects make the practical implementation approximate. This enables importance reweighting of generated samples without retraining the generative model.

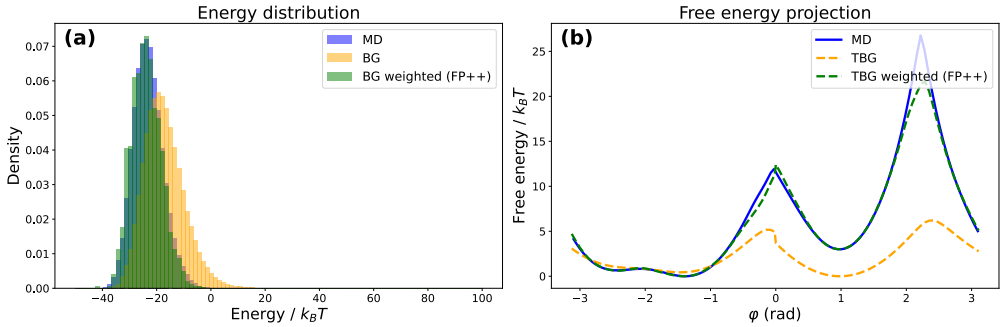


Figure 11: Energy distribution and free energy profiles for the alanine dipeptide system. (a) Energy histograms comparing MD reference, unweighted TBG samples, and FP++-weighted samples. (b) Free energy profiles along the dihedral angle ϕ, computed from MD, unweighted TBG, and FP++-weighted samples. The sample size is 200,000 configurations.

We draw 200,000 independent configurations from the pretrained TBG model (denoted as unweighted samples). For each sample, FP++ estimates the multiplicative determinant correction, which is accumulated in log space to form importance weights $\exp(-W)$. In our implementation, the underlying ODE is solved using a 50-step fourth-order Runge–Kutta (RK4) integrator, and we use the FP3++ estimator with three probes per step. These weights are then used to construct a reweighted distribution ("FP++ weighted").

We evaluate two key thermodynamic statistics:

1. **Energy distribution.** Figure 11(a) shows energy histograms for MD simulations, unweighted TBG samples, and FP++-weighted samples. The unweighted TBG distribution exhibits a clear mismatch with the MD reference, whereas FP++ weighting significantly corrects this discrepancy, closely matching the ground-truth energy distribution.
2. **Free energy profile.** We compute the free energy landscape along the dihedral angle ϕ, the slowest collective variable of alanine dipeptide. Kernel density estimation is used to estimate $P(\phi)$, followed by $F(\phi) = -k_B T \ln P(\phi)$. As shown in Figure 11(b), unweighted samples exhibit biased free energy barriers and incorrect relative basin depths, while FP++-weighted results closely align with MD simulations, including accurate recovery of transition regions.

Overall, these results demonstrate that FP++ enables accurate thermodynamic reweighting of pretrained generative models without retraining, providing a principled correction mechanism for Jacobian-induced distributional bias in molecular simulation.

Table 12: Alanine-dipeptide direct-reweighting diagnostics. Arrows report the metric before and after weighting.

Metric	FP++ direct	TBG official
Valid aligned samples	199885	199871
ESS	3309.496	12854.656
Normalized ESS	0.016557	0.064315
Energy-W2 (unw.→w.)	$6.175459 \rightarrow 0.709800$	$6.714405 \rightarrow 0.794129$
ϕ torsion-W2 (unw.→w.)	$1.438921 \rightarrow 0.009786$	$1.446999 \rightarrow 0.006618$
ψ torsion-W2 (unw.→w.)	$0.752541 \rightarrow 0.046143$	$0.760560 \rightarrow 0.027939$
Normalized weight variance	1.486644×10^{-9}	3.641829×10^{-10}
Normalized weight CV^2	59.397412	14.548529
Log-weight variance	12.548962	11.570547

Tables 12 and 13 summarize the trade-off between computational cost and reweighting quality. Official TBG weighting has higher normalized ESS and lower weight CV^2, whereas FP++ gives

Table 13: Jacobian-computation cost on 100 alanine-dipeptide samples.

Method	Time (s)	s/sample	Alloc. (MB)	Reserved (MB)
FP3++	80.511	0.805	608.22	676.00
Exact Jacobian	569.240	5.692	4276.49	6614.00

slightly lower Energy-W2 and torsion-W2 of comparable magnitude while reducing exact-Jacobian cost: FP3++ is $7.07\times$ faster, with $7.03\times$ lower peak allocated memory and $9.78\times$ lower peak reserved memory. The idealized pass-count ratio is $67/7 = 9.57\times$ because three central-difference probes require six perturbed evaluations in addition to the forward solve; solver and memory overhead account for the smaller observed speedup.

H.9 Reweighting the Ace-AAA-Nme system

We evaluate reweighting on a 42-atom Ace-AAA-Nme system (126 coordinate dimensions) using an ECNF++ generator. The same 2,000 generated samples are reweighted with either the exact Jacobian or FP5++. Results are shown in Table 14; exact-Jacobian weighting yields higher ESS and lower Energy-W2, whereas FP5++ yields lower torsion-joint W2.

Table 14: AL4_ECNF (Ace-AAA-Nme, 300 K) direct-reweighting metrics.

Reweighting	ESS/N	Energy-W2	Torsion-joint W2
Exact Jacobian	0.006896	6.0864	3.1380
FP5++	0.000620	9.4396	2.7036

H.10 Key hyperparameters for reproducibility

Several hyperparameters are critical for reproducing the variance reduction observed in FP and FP++ methods. The exact hyperparameter values and full implementation details are available in the public code repository listed under Software and Data in the main text.

I Impact statement

FP++ provides a general-purpose, determinant-unbiased alternative (for exact inverse-step maps and directional derivatives) to Hutchinson trace estimators for computing Jacobian determinants in continuous generative flows. By reducing variance under the conditions in Appendix C and supporting multi-noise extensions, it enables rigorous thermodynamic inference and Boltzmann reweighting from pretrained generative models. This work advances the use of flow-based generative models in scientific applications, including molecular simulations and high-dimensional inference, without raising immediate ethical concerns.

J Statement on the use of large language models

During the preparation of this manuscript, large language models (LLMs) were used in a limited manner solely for language editing purposes, such as improving clarity, grammar, and academic style. All aspects of the research conception, methodological development, experimental design, analysis of results, and the scientific conclusions presented in this paper were carried out independently by the authors.

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: The abstract summarizes the ideal estimator and tested applications. Section 4.1 and Appendices B–C specify the exact-inverse assumptions, the numerical approximation, and the conditions for variance reduction; the empirical scope is stated in the results and limitations.

   Guidelines:

   - The answer [N/A] 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 [N/A] 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: The limitations paragraph and Appendix H.3 discuss finite-difference, roundoff, solver-inversion, covariance-condition, and scope limitations.

   Guidelines:

   - The answer [N/A] 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: Appendix B states the exact-inverse assumptions for determinant unbiasedness. Appendix C gives the covariance decomposition and sufficient conditions rather than claiming universal variance reduction.

Guidelines:

- The answer [N/A] 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: Section 4.1, Appendix A, and Appendix H.10 specify the determinant estimator, particle counts, annealing schedules, resampling rule, mutation steps, and key numerical settings.

Guidelines:

- The answer [N/A] 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: The public repository linked in Section 6 provides the implementation, environment requirements, and training, MC, and SMC usage examples; externally released molecular data and pretrained models are cited in the paper.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.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://neurips.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) necessary to understand the results?

   Answer: [Yes]

   Justification: Training and evaluation details for the 1000D GMM and Chignolin mutant are specified in Section 5.2 and Appendix H.10, including annealing schedules, MCMC steps, and force field parameters.

   Guidelines:

   - The answer [N/A] 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: Tables 2, 3, and 4 report means and standard deviations over ten independent runs for FP/FP++ and Hutchinson estimators. The reduced-particle, single-run 1000D BFJacob reference is explicitly identified. Standard deviations quantify run-to-run variability rather than constituting a formal significance test; representative-run diagnostics are labeled separately.

Guidelines:

- The answer [N/A] 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: [No]

   Justification: We report ODE-equivalent passes, per-run wall-clock times, and the alanine-dipeptide runtime and peak-memory benchmark, but the complete hardware inventory and total project-wide compute (including exploratory runs) are not fully reported for every experiment.

   Guidelines:

   - The answer [N/A] 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: Our work adheres to the NeurIPS Code of Ethics regarding scientific integrity and responsible data usage in molecular modeling research.

   Guidelines:

   - The answer [N/A] 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: [Yes]

    Justification: The impact statement in Appendix I discusses scientific benefits for molecular sampling and high-dimensional inference; the authors do not identify immediate ethical concerns for these applications.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The proposed method and datasets (GMM and proteins) are scientific in nature and do not pose a high risk of malicious misuse or dual-use.

    Guidelines:

    - The answer [N/A] 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: [No]

Justification: The sources of the force-field parameters, molecular data, pretrained flows, and comparison models are credited. However, a complete asset-by-asset inventory of licenses and terms of use is not explicitly provided; we therefore do not claim that all assets have an academic or MIT license.

Guidelines:

- The answer [N/A] 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: [Yes]

Justification: The FP++ implementation is released in our repository alongside environment requirements and training, MC, and SMC usage examples.

Guidelines:

- The answer [N/A] 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: [N/A]

Justification: This research does not involve crowdsourcing or human subjects.

Guidelines:

- The answer [N/A] 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: [N/A]

    Justification: Our study is limited to physical simulations and mathematical modeling; no human subject research was performed.

    Guidelines:

    - The answer [N/A] 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 rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs were not an important, original, or non-standard component of the core research method; any use was limited to writing, editing, or formatting, for which the NeurIPS question states that a declaration is not required.

    Guidelines:

    - The answer [N/A] 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 in the NeurIPS handbook for what should or should not be described.