Liouville Flow Importance Sampler

Yifeng Tian [*1] **Nishant Panda** [1] **Yen Ting Lin** [*1]

Abstract

We present the Liouville Flow Importance Sampler (LFIS), an innovative flow-based model for generating samples from unnormalized density functions. LFIS learns a time-dependent velocity field that deterministically transports samples from a simple initial distribution to a complex target distribution, guided by a prescribed path of annealed distributions. The training of LFIS utilizes a unique method that enforces the structure of a derived partial differential equation to neural networks modeling velocity fields. By considering the neural velocity field as an importance sampler, sample weights can be computed through accumulating errors along the sample trajectories driven by neural velocity fields, ensuring unbiased and consistent estimation of statistical quantities. We demonstrate the effectiveness of LFIS through its application to a range of benchmark problems, on many of which LFIS achieved state-of-the-art performance.

1. Background

We are interested in sampling a hard-to-sample distribution ν in a continuous state space[1] $x \in \mathbb{R}^D$, given its unnormalized probability density function $\tilde{\nu}(x)$ (assuming its existence, i.e., ν is dominated by the Lebesgue measure) up to a normalization constant $\mathcal{Z}$. We are also interested in estimating $\log \mathcal{Z}$, which is the log-marginal likelihood of a model and is a useful quantity for Bayesian model selection (Kass & Raftery, 1995; Green, 1995; Llorente et al., 2023). This problem arises in vast scientific domains, including statistical physics (Faulkner & Livingstone, 2023), molecular dynamics (Hénin et al., 2022), and diverse Bayesian inference and model selection problems, for example in computational and systems biology (Hines, 2015; Wilkinson, 2007), astronomy (Sharma, 2017), and political sciences (Jackman, 2000; McCartan & Imai, 2023).

1.1. Related Work

Numerous Monte Carlo (MC) techniques have been devised to address this challenge over the past 70 years. Some notable examples of these MC methods include the landmark Metropolis Markov chain MC (MCMC, (Metropolis et al., 1953)), Hybrid or Hamiltonian MC (HMC, Duane et al. (1987); Neal (2012)), and the Zig-Zag Sampler (Bierkens et al., 2019). These approaches are often limited to low-dimensional problems due to slow convergence rate, and problems with a single mode as the samplers are prone to be trapped at a local mode. Annealed Importance Sampler (AIS, Neal (2001)) and Sequential Monte Carlo (SMC, (Del Moral et al., 2006)) were developed for sampling challenging multi-modal distributions and are considered the state-of-the-art MC methods for such problems. Variational Inference (VI, Wainwright & Jordan (2008)) is an alternative approach, which turns the sampling problem into an optimization one by fitting an easy-to-sample parametric distribution to ν, often by minimizing the reverse (or exclusive) Kullback–Leibler (KL) divergence. The choice of the reverse KL is commonly thought to be due to computational convenience but can be validated by an information theoretical argument of optimal information processing of Bayes' theorem (Zellner, 1988). For VI, it is popular to adopt mean-field approximation (Wainwright & Jordan, 2008) or a Normalizing Flow (VI-NF, Rezende & Mohamed (2015)) as the modeling parametric distribution. More recently, a plethora of neural network (NN) based algorithms, for example, neural network gradient HMC (Li et al., 2019), Annealed Flow Transport Monte Carlo (AFTMC, Arbel et al. (2021a)), Path-Integral Sampler (PIS, Zhang & Chen (2022)), adaptive MC with NF Gabrié et al. (2022), Continual Repeated AFTMC (CR-AFTMC, Matthews et al. (2022)), and Denoising Diffusion Sampler (DDS, (Vargas et al., 2023a)). For estimating the marginalized likelihood, simple MCMC methods require augmented algorithms (e.g., Green (1995)) or additional processing of the derived samples (e.g., Robert & Wraith (2009); Pajor (2017)). SIS and SMC can estimate the marginalized likelihood and are considered the gold-

[*]Equal contribution [1]Information Sciences Group (CCS-3), Computational and Statistical Sciences Division, Los Alamos National Laboratory, Los Alamos, NM 87545, USA. Correspondence to: Yifeng Tian <yifengtian@lanl.gov>, Yen Ting Lin <yenting1@lanl.gov>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

[1]We can also consider a subspace $x \in \mathcal{E} \subseteq \mathbb{R}^D$.

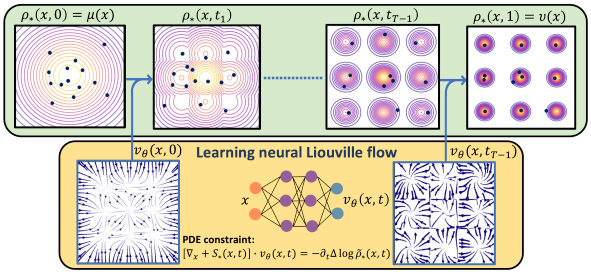

Figure 1. A schematic diagram demonstrating the workflow of the Liouville Flow Importance Sampler.

standard MC estimates. Among the NN-based algorithms, AFTMC, CR-AFTMC, PIS, and DDS are capable of estimating the marginalized likelihood. Llorente et al. (2023) provides a more comprehensive review of methods estimating the marginalized likelihood.

1.2. Summary of our contribution

We propose *Liouville Flow Importance Sampler*, a novel pure flow-based method with which we achieve sampling ν by (1) constructing a time-dependent target distribution[2] $\rho_*(t)$ connecting an easy-to-sample distribution μ and our target ν, where $t \in [0,1]$ is a fictitious time, i.e., $\rho_*(0) = \mu$ and $\rho_*(1) = \nu$; (2) solving a time-dependent velocity field $v(x,t) : \mathbb{R}^D \times [0,1] \to \mathbb{R}^D$ satisfying the following Generalized Liouville Equation (GLE, Gerlich (1973)),

$$\partial_t \rho_*(x,t) = -\nabla_x \cdot [v(x,t)\rho_*(x,t)]\,, \rho_*(x,0) = \mu(x) \quad (1)$$

(3) drawing N i.i.d. samples $x_0^{(i)}$, $i = 1 \dots N$ from μ, then using the solved velocity field to evolve the samples from $t = 0 \to 1$ by the ordinary differential equation

$$\dot{x}^{(i)}(t) = v\left(x^{(i)}(t), t\right),\ x^{(i)}(0) = x_0^{(i)}. \quad (2)$$

The evolved samples at time $t = 1$ are samples of ν because GLE ensures that the probability density of independently evolving samples is the density function in Eq. (1), i.e., $x^{(i)}(t) \sim \rho_*(t)$, and because $\rho_*(1) = \nu$. The schematic Fig. 1 illustrates the workflow of our proposal.

We summarize our novel contributions in this study, leaving a detailed discussion of the differences between our proposition and existing methods in the Results and Discussion section 4:

• We introduce a pure flow-based model that deterministically transports samples from an initial distribution μ to a target distribution ν, relying exclusively on the flow mechanism and consistent with a prescribed path of annealed distributions, $\tilde{\rho}_*(t)$. This innovation streamlines the modeling process, eliminating the complex tuning of meta and algorithmic parameters that is typical in hybrid models leveraging both flow and Monte Carlo methods.

• We propose an original approach to solving a key equation (Eq. (5) below), which $v(x,t)$ must satisfy to ensure the samples move consistently with the specified time-dependent and unnormalized density function $\tilde{\rho}_*(x,t)$. We convert the problem to an equation-based machine learning task where samples drawn from μ are used to train an NN that models the velocity field at a specific time. Trained NNs are used to generate new samples for training the velocity field at a later time. This recursive training of a series of NNs facilitates end-to-end sample transportation from μ to ν.

• We demonstrate a novel derivation that, although finite NNs do not perfectly learn the solution of Eq. (5), the accumulated error induced along the sample trajectory can be used as sample weights, allowing for unbiased and consistent estimation of statistical quantities.

• The proposed method, while having a simpler setup compared to existing methods, achieves state-of-the-art performance across a variety of standard benchmarks.

2. Liouville Flow Importance Sampler

2.1. Governing equation of the velocity field

Our proposition hinges on the key research question: Given an unnormalized time-dependent density function $\tilde{\rho}_*(x,t) = \mathcal{Z}(t)\rho_*(x,t)$, up to an unknown normalization constant $\mathcal{Z}(t) := \int \tilde{\rho}_*(x,t)\,\mathrm{d}x$, how do we solve a time-dependent velocity field $v(x,t)$?

We first ask the question: what is the equation a velocity field $v(x,t)$ must satisfy in terms of only the unnormalized density function $\tilde{\rho}_*(x,t)$ (instead of the inaccessible full density function $\rho(x,t)$ in Eq. (1))? On the one hand, using GLE (1) and the chain rule, it is straightforward to establish

$$\partial_t \log \rho_*(x,t) = -\nabla_x \cdot v\,(x,t) - v\,(x,t) \cdot S_*\,(x,t)\,, \quad (3)$$

where $S_*\,(x,t) := \nabla_x \log \rho_*\,(x,t) \equiv \nabla_x \log \tilde{\rho}_*\,(x,t)$ is the score function of the target density at time t. Here, we assume that the derivative of the log-density exists and can be evaluated. On the other hand, applying ∂_t to $\log \rho_*\,(x,t) = \log \tilde{\rho}_*\,(x,t) - \log \mathcal{Z}\,(t)$ leads to

$$\partial_t \log \rho_*(x,t) = \partial_t \log \tilde{\rho}_*(x,t) - \langle \partial_t \log \tilde{\rho}_*(\cdot,t)\rangle_*\,, \quad (4)$$

where we used a common notation in statistical physics, $\langle \partial_t \log \tilde{\rho}_*(\cdot,t)\rangle_* := \mathbb{E}_{x' \sim \rho_*(\cdot,t)}\left[\partial_t \log \tilde{\rho}_*(x',t)\right]$. This term emerged from the time-derivative of the unknown normalization constant $\mathcal{Z}(t)$. Equating Eqs. (3) and (4), we obtained

[2] We use $\rho_*(t)$ to denote time-dependent distribution and $\rho_*(x,t)$ and $\tilde{\rho}_*(x,t)$ to denote its normalized and unnormalized density function respectively.

the following equation:

$$[\nabla_x + S_*(x,t)] \cdot v(x,t) = -\partial_t \delta \log \tilde{\rho}_*(x,t), \quad (5a)$$

$$\delta \log \tilde{\rho}_*(x,t) := \log \tilde{\rho}_*(x,t) - \langle \log \tilde{\rho}_*(\cdot,t) \rangle_* . \quad (5b)$$

The below theorem, whose proof is given in Appendix A, states the existence of a solution to Eq. (5).

Theorem 2.1. *Suppose $\partial_t \delta \log \tilde{\rho}_*(x,t)$ is square-integrable and the divergence of target score function is absolutely bounded i.e., $\exists M < \infty$ s.t. $|\nabla_x \cdot S_*(x,t)| < M$ almost everywhere for all t. Then, there exists a solution of Eq. (5), but the solution is not unique in $D \geq 2$ dimensions.*

We remark that the non-uniqueness of the solution does not pose an issue for our objectives. Any solution of $v(x,t)$ satisfying Eq. (5), once obtained, has an equivalent effect of transporting the samples.

2.2. Learning velocity field for sampling

It is beyond the authors' knowledge how to solve Eq. (5) accurately in an arbitrary dimension D. As such, instead of solving the equation, we propose to *learn* a velocity field that approximately satisfies Eq. (5) by the following procedure. Suppose we could generate samples $x^{(i)}$ at time t from $\rho_*(t)$. Both $S_*(x^{(i)},t)$ and $\partial_t \log \tilde{\rho}_*(x^{(i)},t)$ can be evaluated, and $\langle \partial_t \log \tilde{\rho}_*(\cdot,t) \rangle_*$ can be estimated by Monte Carlo, i.e., $(1/N)\sum_{i=1}^N \partial_t \log \tilde{\rho}_*(x^{(i)},t)$. Next, we use an NN to model the velocity field at this specific time t, i.e. $v_\theta(x,t) = \mathrm{NN}(x;\theta,t)$ with NN parameters θ. Using automatic differentiation, the divergence term, $\nabla_x \cdot v_\theta(x^{(i)},t)$ can be evaluated for each sample. Replacing the solution $v(x,t)$ in Eq. (5) with the neural velocity field $v_\theta(x^{(i)},t)$, our objective is to minimize the discrepancy of the LHS and the RHS in Eq. (5) over the samples at time t,

$$\theta_* = \mathrm{argmin}_\theta \sum_{i=1}^N \varepsilon^2 \left(x^{(i)}; \theta \right), \quad (6a)$$

$$\varepsilon(x;\theta) := [\nabla_x + S_*(x)] \cdot v_\theta(x) + \partial_t \delta \log \tilde{\rho}_*(x), \quad (6b)$$

where we have suppressed the t-dependence of S_*, v_θ, and $\tilde{\rho}$ for brevity. After the learning, we use the learned velocity field to evolve the sample to a later time $t+\Delta t$ by an explicit Euler scheme, $x^{(i)} \leftarrow x^{(i)} + v_{\theta_*}\left(x^{(i)},t\right) \times \Delta t$, $\Delta t \ll 1$. In the ideal scenario that the NN learns the solution of Eq. (5) and the error induced by the time discretization is negligible, the GLE (1) ensures that the transported samples $\sim \rho_*(t + \mathrm{d}t)$. In turn, the samples can be used to learn the velocity field at $t + \Delta t$. Repeating the process, the samples can be recursively evolved from $t = 0 \to 1$.

Our proposition of learning the velocity field using equation Eq. (5) is similar to the Physics Informed Machine Learning (Raissi et al., 2019; Karniadakis et al., 2021). Moreover, our proposition can be considered as a self-learning framework because samples are not labeled, and because we can always generate more samples from μ and evolve them by previously trained $v(x,s)$ to $t > s$, for training $v(x,t)$. We note that Eq.(5) has been explicitly presented in Vaikuntanathan & Jarzynski (2008), Heng et al. (2021), and Arbel et al. (2021a); a more detailed discussion is provided in Sec. 4.

We remark that our methodology significantly differs from the Flow Matching approach (Lipman et al., 2023; Tong et al., 2024). Flow Matching trains a continuous-time normalizing flow by matching it to a *known* velocity field. This technique is specifically designed for training Generative Models using samples of a data distribution. In contrast, our task involves learning the normalizing flow using a prescribed path of annealed distributions, $\tilde{\rho}_*(x,t)$. In this scenario, we do not have access to the samples of the target distribution nor a target velocity field for matching: the problem would have been solved if we had the samples of the target distribution, and we could have simply driven the samples by a known target velocity field. Instead, our knowledge is limited to the fact that the velocity field at each of the samples must satisfy Eq. (5). Our goal is to solve for the unknown $v(x,t)$ in Eq. (5), rather than attempting to match a target flow.

2.3. Neural network as an importance sampler

Our proposition described in Sec. 2.2 works *only if* the NN is expressive enough to accurately reproduce the true solution satisfying Eq. (5). Without the assumption, transported samples would not be representative samples at $t + \mathrm{d}t$ for training v_θ.

More specifically, let us consider a sub-optimal velocity field $v_{\theta_*}(x,t)$ which does not satisfy Eq. (5), $\forall x \in \mathbb{R}^D$. This is very likely because the training of an NN is not perfect, or because we may not have enough computational resources to train an expressive enough network to fully solve Eq. (5). Using the trained NNs to transform the initial distribution μ, the induced distribution $\rho_\theta(t)$ still satisfies a GLE $\partial_t \rho_\theta(x,t) = -\nabla_x [v(x,t)\rho_\theta(x,t)]$ but $\rho_\theta(t)$ is no longer be the same as $\rho_*(t)$. In this case, the RHS of Eq. (3) becomes $-\nabla_x \cdot v_\theta(x,t) - S_\theta(x,t) \cdot v_\theta(x,t)$, where $S_\theta := \nabla_x \log \rho_\theta(x,t)$, and consequently we could not establish Eq. (5). In addition, we will no longer have samples to estimate unbiasedly the term $\langle \partial_t \log \tilde{\rho}_*(\cdot,t) \rangle_*$ in Eq. (5). Our proposition outlined in Sec. 2.2 appears to lose its theoretical grounding.

Fortunately, a more careful derivation presented in Appendix B revealed that we can use the trained, albeit imperfect, neural networks to transport the samplers as an importance sampler. We provide key equations here, leaving detailed derivation and proof in Appendices B and C.

We first write the generic solution of Eq. (2):

$$x^{(i)}(t) = x_0^{(i)} + \int_0^t v_\theta \left(x(s; x_0^{(i)}), s \right) \, \mathrm{d}s, \tag{7}$$

where v_θ is an NN-modeled velocity field and $x_0^{(i)} \sim \mu$ is the initial condition. Below, when the context is clear, we will suppress the sample superscript (i) and write $x(t)$ and x_0 for brevity. We stress that the sample trajectory implicitly depends on the initial condition x_0, which is the only randomness; once x_0 is drawn, $x(t)$ is a deterministic trajectory. Then, the probability density of the modeling distribution along this trajectory can be computed (see Appendix B):

$$\log \rho_\theta(x(t), t) = \log \mu(x_0) - \int_0^t \nabla \cdot v_\theta \left(x(s), s \right) \, \mathrm{d}s. \tag{8}$$

Treating ρ_θ as an importance sampler, interestingly, the unnormalized log-weights of the samplers can be shown to be the error accumulated along each of the sample trajectories:

$$\log \tilde{w} \left(t; x_0 \right) = \int_0^t \varepsilon \left(x \left(s; x_0 \right) \right) \, \mathrm{d}s, \tag{9}$$

where ε is defined in Eq. (6b). With N initial samples $x_0^{(i)} \sim \mu$, $i = 1 \dots N$, we can obtained the normalized weights (Liu, 2008; Tokdar & Kass, 2010)

$$w_i(t) := \tilde{w} \left(t, x_0^{(i)} \right) / \sum_{j=1}^N \tilde{w} \left(t, x_0^{(j)} \right). \tag{10}$$

which can be used to estimate unbiasedly asymptotically (i.e. unbiased as $N \to \infty$)

$$\left\langle \partial_t \log \tilde{\rho}_*(\cdot, t) \right\rangle_* \approx \sum_{i=1}^N w_i(t) \partial_t \log \tilde{\rho}_* \left(x^{(i)}(t), t \right), \tag{11}$$

which can be used for learning $v(x, t)$ without an accurate solution of Eq. (5). Furthermore, a derivation akin to Gelman & Meng (1998) revealed that we can unbiasedly asymptotically estimate the logarithm of the marginalized likelihood $\log \mathcal{Z}(t)$,

$$\widehat{\log \mathcal{Z}(t)} \approx \int_0^t \sum_{i=1}^N w_i(s) \, \partial_s \log \tilde{\rho}_* \left(x^{(i)}(s), s \right) \mathrm{d}s, \tag{12}$$

which can be computed on-the-fly as samples are transported from $t = 0 \to 1$. Theorem 2.2, whose proof is given in Appendix C, articulates the mathematical statements about the asymptotically unbiased estimator.

Theorem 2.2. *For any $t \in [0, 1]$, suppose the $\partial_t \log \tilde{\rho}_*(x, t)$ is an integrable function with respect to the target distribution $\rho_*(t)$. Let $\{x_0^{(i)}\}_{i=1}^N$ be a finite set of initial samples drawn from the initial distribution μ and consider the following finite sum for each time $s \in (0, t)$ given by*

$$W_N(s) = \sum_{i=1}^N w_i(s) \, \partial_t \log \tilde{\rho}_* \left(x^{(i)}(s), s \right) \tag{13}$$

where $w_i(s)$ is the dynamic sample weight along the trajectory given by the evolution of $x_0^{(i)}$ as defined in Eqs. (9) and (10). Suppose the following conditions hold: (1) the neural velocity field induced distribution $\rho_\theta(t)$ dominates the target distribution $\rho_*(t)$ and, (2) both the dynamic weights $\tilde{w}(t; x_0)$ and $\partial_t \log \tilde{\rho}$ are absolutely bounded, i.e there exists constants M_1, M_2 such that $|\tilde{w}(t; x_0)| < M_1$ and $|\partial_t \log \tilde{\rho}_*(x, t)| < M_2$. Then, the marginalized likelihood can be estimated unbiasedly asymptotically (i.e. unbiased as $N \to \infty$) and persistently

$$\int_0^t W_N(s) \, ds \xrightarrow{a.s.} \log \mathcal{Z}(t) \text{ as } N \to \infty.$$

While the above theorem provides the unbiased estimation of $\log \mathcal{Z}$ directly, i.e., $\widehat{\log \mathcal{Z}}$, many published methods focus on unbiasedly estimating $\mathcal{Z}$ directly, i.e., $\hat{\mathcal{Z}}$, but report $\log \hat{\mathcal{Z}}$. To make a fair comparison, we established a similar estimator for LFIS in the following theorem

Theorem 2.3. *For any $t \in [0, 1]$, suppose $\partial_t \log \tilde{\rho}_*(x, t)$ is an integrable function with respect to the target distribution $\rho_*(t)$. Let $\epsilon(x; \theta)$ be a modified error function that is analogous to Eq. (6b) as*

$$\epsilon(x; \theta) := \left[\nabla_x + S_*(x) \right] \cdot v_\theta(x) + \partial_t \log \tilde{\rho}_*(x), \tag{14}$$

and the modified unnormalized importance weights that are analogous to Eq. (9):

$$\log \varpi(t; x_0) = \int_0^t \epsilon \left(x(s; x_0) \right) ds. \tag{15}$$

If the same conditions as in Theorem 2.2 hold, the marginalized likelihood $\mathcal{Z}$ can be unbiasedly estimated:

$$\hat{\mathcal{Z}}(t) = \frac{1}{N} \sum_{i=1}^N e^{\log \varpi(t; x_0)}. \tag{16}$$

The proof of the above estimate is given in Appendix C. In the rest of the main text, we focus on $\log \hat{\mathcal{Z}}$ to ensure a fair comparison between different methods and report $\widehat{\log \mathcal{Z}}$ in the Appendix (Table 11).

2.4. Choice of $\tilde{\rho}_*(x, t)$ and the schedule function $\tau(t)$

Our proposition requires an unnormalized time-dependent target density function $\tilde{\rho}_*(x, t)$. In this manuscript, we primarily consider two types of applications: (1) sampling a

Algorithm 1 Liouville Flow Importance Sampler

procedure GENERATESAMPLES(k, n, $\theta_*^{(0:k-1)}$, optional $\langle \partial_t \log \tilde{\rho}_* \rangle_*^{(0:k-1)}$)
▷ Generate n samples at a target time k/T using previously trained $v_{\theta_*^{(j)}}$ and optionally provided $\langle \partial_t \log \tilde{\rho}_* \rangle_*^{(0:k-1)}$
$x_i \sim \mu$, $\delta_i \leftarrow 0$, $i = 1 \dots n$ ▷ Drawing samples from the initial distribution
for $\ell = 0 \dots k-1$ **do** ▷ Evolve samples using previously trained fields v_{θ_*} and estimated $\langle \partial_t \log \tilde{\rho}_* \rangle_*$
If $\langle \partial_t \log \tilde{\rho}_* \rangle_*^{(\ell)}$ is not provided **then** $\langle \partial_t \log \tilde{\rho}_* \rangle_*^{(\ell)} \leftarrow \sum_{i=1}^n \exp(-\delta_i) \partial_t \log \tilde{\rho}_* \left(x_i, \ell/T\right) / \left(T \sum_{j=1}^n \exp(-\delta_j)\right)$
$\delta_i \leftarrow \delta_i + \left\{ \left[\nabla_x + S_* \left(x_i, \ell/T\right)\right] \cdot v_{\theta_*^{(\ell)}} \left(x_i\right) + \partial_t \log \tilde{\rho}_* \left(x_i, \ell/T\right) - \langle \partial_t \log \tilde{\rho}_* \rangle_*^{(\ell)} \right\} / T$ ▷ Error accumulation
$x_i \leftarrow x_i + v_{\theta_*^{(\ell)}} \left(x_i\right) / T$ ▷ Transporting samples
$w_i \leftarrow \exp(-\delta_i) / \sum_{j=1}^n \exp(-\delta_j)$ ▷ Weight computation
return $\{x_i, w_i\}_{i=1}^N$
procedure LEARNING( ) ▷ Learning the flow
for $k = 0 \dots T-1$ **do**
$\{x_i, w_i\}_{i=1}^N = \text{GenerateSamples}\left(k,\ N,\ \theta_*^{(0:k-1)},\ \langle \partial_t \log \tilde{\rho}_* \rangle_*^{(0:k-1)}\right)$ ▷ Samples for estimating $\langle \partial_t \log \tilde{\rho}_* \rangle_*^{(k)}$
$\langle \partial_t \log \tilde{\rho}_* \rangle_*^{(k)} \leftarrow \sum_{i=1}^N w_i \partial_t \log \tilde{\rho}_* \left(x_i, k/T\right)$ ▷ Store the estimated $\langle \partial_t \log \tilde{\rho}_* \rangle_*^{(k)}$
while Training criteria are not met **do** ▷ Trianing the NN
$\{x_i, w_i\}_{i=1}^B = \text{GenerateSamples}\left(k,\ B,\ \theta_*^{(0:k-1)},\ \langle \partial_t \log \tilde{\rho}_* \rangle_*^{(0:k-1)}\right)$ ▷ Batch (B) samples for training
$\varepsilon_i(\theta) \leftarrow \left[\nabla_x + S_* \left(x_i, k/T\right)\right] \cdot v_\theta \left(x_i\right) + \partial_t \log \tilde{\rho}_*(x_i, k/T) - \langle \partial_t \log \tilde{\rho}_* \rangle_*^{(k)}$ ▷ Sample-by-sample error
$\theta \leftarrow \text{GradientDecentStep}\left(\theta, \nabla_\theta \left(\sum_{i=1}^B \varepsilon_i^2(\theta)/B\right)\right)$
$\theta_*^{(k)} \leftarrow \theta$ ▷ Store trained NN parameters
procedure SAMPLING ▷ Generate S samples of ν after learning the flow
return $\text{GenerateSamples}\left(T,\ S,\ \theta_*^{(0:T-1)}\right)$

distribution ν, given its unnormalized density function $\tilde{\nu}(x)$ and (2) Bayesian posterior sampling, given a prior density function $\pi(x)$ and a likelihood function $L(x)$. Motivated by AIS (Neal, 1996; 2001) and SMC (Del Moral et al., 2006), we consider $\tilde{\rho}_*(x,t) := \mu^{1-\tau(t)}(x)\, \tilde{\nu}^{\tau(t)}(x)$ for application (1) and $\tilde{\rho}_*(x,t) := L^{\tau(t)}(x)\pi(x)$ for application (2), where $\tau(t)$ is a monotonic function transforming time t, satisfying $\tau(0) = 0$ and $\tau(1) = 1$. We term $\tau(t)$ as the *schedule function*. The corresponding key quantities are

$$\partial_t \log \tilde{\rho}_*(x,t) = \left[\log \tilde{\nu}(x) - \log \mu(x)\right] \frac{\mathrm{d}\tau\left(t\right)}{\mathrm{d}t},$$
$$S_*(x,t) = \left(1 - \tau\left(t\right)\right) \nabla_x \log \mu(x) + \tau\left(t\right) \nabla_x \log \tilde{\nu}(x)$$

for type-1 application, and

$$\partial_t \log \tilde{\rho}_*(x,t) = \log L(x) \frac{\mathrm{d}\tau\left(t\right)}{\mathrm{d}t},$$
$$S_*(x,t) = \tau\left(t\right) \nabla_x \log L(x) + \log \pi(x)$$

for application type-2. We remark that the general learning and sampling procedures of LFIS do not depend on the particular choice of $\tilde{\rho}_*(t)$.

We term our proposition in Sec. 2.2 with the importance sampling in Sec. 2.3 and the choices of $\tilde{\rho}_*(x,t)$ in Sec. 2.4 as the *Liouville Flow Importance Sampler (LFIS)*.

3. Numerical Experiments

In this section, we present numerical experiments comparing the proposed LFIS with other state-of-the-art approaches using NNs, including AFTMC, PIS, and DDS. We left out VI-NF as its inferior performance has been established in Arbel et al. (2021a). We also optimized an SMC to provide a reference of the performance of a state-of-the-art sampling algorithm without NNs; see Appendix D.10.

We use a set of NNs to model the temporally discretized velocity field $v(x,t)$, at $t = 0, 1/T, 2/T, \dots, 1$. Here, T is the total number of time steps, which plays the same role as the number of tempering scales (or "temperatures") in AIS, AFT, or SMC. We investigated $T = 32$, 64, 128, and 256. Results with $T = 256$ of all the methods are presented in the main manuscript, and others are presented in Appendix D.9 for completeness. We chose a cosine schedule function (see Sec. 2.4) based on the results of a smaller-scale analysis presented in Appendix D.5. At each discrete time step, we used a separate feed-forward NN with a similar structure as in Vargas et al. (2023a) and Zhang & Chen (2022) (two hidden layers, each of which has 64 nodes) to model the discrete-time velocity field. Except for the first ($t = 0$) NN, which was initialized randomly, we instantiated the NN at $t = k/T$ using the weights of the trained NN at the

previous time $t = (k-1)/T$ to amortize the training cost. We initialized the weights of the last layer of the NN to be zero, which was observed to expedite the training process empirically. The divergence of the flow field and the score function can either be computed theoretically or by using the `autograd` function in `PyTorch`.

Algorithm 1 provides a more detailed description of the implementation of LFIS. Samples are drawn from either standard isotropic Gaussian distribution for type-1 applications, or from the prior distribution for type-2 applications (see Sec. 2.4). The samples are transported by the previously learned velocity fields to a specific time step, which will then be used for learning the velocity field at the next time step. Our experiments suggested that the learning performs better with the following implementations. (1) A large number of samples ($N = 5 \times 10^4$) are used to estimate $\langle \partial_t \log \tilde{\rho}_* (t) \rangle_*$ at a specific time. For computational efficiency, we only draw a fixed batch at each target time to estimate this quantity once. (2) A small batch of samples ($B = 300$-10000 depending on applications) are drawn for each gradient descent step.

After training, we performed 30 independent samplings (each with $S = 2000$ samples, different from those used in training) using the learned velocity fields. Following the recent series of papers (Arbel et al., 2021a; Zhang & Chen, 2022; Vargas et al., 2023a), we focus on assessing sample qualities by the estimation of log normalization constant $\log \mathcal{Z}$. For low dimensional ($D \leq 10$) problems, we also compare the sliced Wasserstein distance W_p (Bonneel et al.) to ground-truth samples with $p = 2$ as an additional metric. For all the methods with importance sampling, we used the weighted samples for computing these statistical quantities.

3.1. Testing problems

Mode-separated Gaussian mixture (type-1): An illustrative model of $D = 2$ Gaussian mixture distribution with separated modes. We consider a similar challenging distribution as in PIS (Zhang & Chen, 2022): a mixture of nine Gaussian distributions centered at the grid $\{-1, 0, 1\}^2$, and each Gaussian has variance 0.012.

Funnel distribution (type-1): A challenging ten-dimensional distribution proposed by Neal (2003) is commonly used for testing samplers. The formulation of the funnel distribution is:

$$x_0 \sim \mathcal{N}(\mu = 0, \sigma^2 = 9), \tag{19a}$$

$$x_{1:9}|x_0 \sim \mathcal{N}(\boldsymbol{\mu} = \mathbf{0}, \boldsymbol{\Sigma} = e^{x_0}\mathbf{I}) \tag{19b}$$

Log Gaussian Cox Process (type-2): The Log-Gaussian Cox Process (LGCP) is a commonly used model for the analysis of spatial point pattern data and is designed for modeling the positions of Findland pine saplings (Møller et al., 1998). The LGCP is a hierarchical combination of a Poisson process and a Gaussian Process prior, which can be naturally framed as a Bayesian problem. Here we use the variant of LGCP on a 40×40 grid, resulting a $D = 1600$ sampling problem. The target posterior density is:

$$\lambda(\mathbf{x}) \sim \mathcal{N}(\mathbf{x}; \mu, K) \prod_i \exp(x_i y_i - \alpha e^{x_i}). \tag{20}$$

To frame the LGCP as a Bayesian posterior sampling problem, we treat the Gaussian Process as the prior distribution and the Poisson process as the likelihood function:

$$\pi(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \mu, K), L(\mathbf{x}) = \prod_i \exp(x_i y_i - \alpha e^{x_i}). \tag{21}$$

Logistic regression (type-2): Here we consider the Bayesian logistic regression with the prior distribution $\pi(\mathbf{x}) = \mathcal{N}(0, I)$, and the logistic regression model $P(y_i) = \text{Bernoulli}(\text{sigmoid}(\mathbf{x}^T \cdot \mathbf{u}_i))$. The Bayesian inference of the logistic regression model is performed on the Ionosphere dataset with $D = 35$ and the Sonar dataset with $D = 61$.

Latent space of Variational Autoencoder (type-2): In this experiment, we investigate sampling in the latent space of a pre-trained Variational Autoencoder (VAE) on the binary MNIST dataset. The posterior distribution in the latent space is denoted as the combination of a Gaussian prior of latent variable $\mathbf{z}$ and the decoder $p_\theta(\mathbf{x}|\mathbf{z})$.

Table 1. Results of type-1 problems. The best model (in bold font) in $\log \hat{\mathcal{Z}}$ is determined by its deviation from the theoretical value, 0. The best model in Wasserstein-2 distance to the ground-truth samples is determined by the smallest value. The best model in effective sample size (EES) is determined by its deviation from the optimal value, 1.

	Model	MG ($D = 2$)	Funnel ($D = 10$)
$\log \hat{\mathcal{Z}}$	SMC	-1.28 $\pm$ 0.01	-0.12 $\pm$ 0.06
	LFIS	**-0.0002 $\pm$ 0.004**	**-0.07 $\pm$ 0.003**
	DDS	-0.31 $\pm$ 0.43	-0.31$\pm$ 0.12
	PIS	0.0035 $\pm$ 0.02	-1.14 $\pm$ 0.13
	AFT	2.43 $\pm$ 0.05	-0.11 $\pm$ 0.68
W_2	SMC	0.066 $\pm$ 0.017	6.07 $\pm$ 1.52
	LFIS	**0.054 $\pm$ 0.014**	**5.57 $\pm$ 1.52**
	DDS	0.368 $\pm$ 0.23	6.54 $\pm$ 1.44
	PIS	0.061 $\pm$ 0.015	6.40 $\pm$ 1.49
	AFT	0.10 $\pm$ 0.042	6.20 $\pm$ 1.41
ESS	SMC	0.99 $\pm$ 0.003	0.99 $\pm$ 0.005
	LFIS	**0.97 $\pm$ 0.001**	**0.97 $\pm$ 0.063**
	DDS	0.012 $\pm$ 0.009	0.163 $\pm$ 0.086
	PIS	0.59 $\pm$ 0.038	0.12 $\pm$ 0.066
	AFT	0.66 $\pm$ 0.185	0.76 $\pm$ 0.277

Table 2. $\log \hat{\mathcal{Z}}$ estimation of the Bayesian problems. As the ground-truth value is not known, SMC (with 1024 scales) results are considered as gold standard. Models that fall within the statistical margin of error relative to the gold-standard values are highlighted in bold font.

Model	LGCP ($D = 1600$)	Ionosphere ($D = 35$)	Sonar ($D = 61$)	VAE ($D = 30$)
SMC (1024)	506.96 $\pm$ 0.24	-111.61 $\pm$ 0.03	-108.38 $\pm$ 0.02	-110.16 $\pm$ 0.41
SMC (256)	**506.77 $\pm$ 0.68**	**-111.62 $\pm$ 0.05**	**-108.39 $\pm$ 0.04**	**-110.20 $\pm$ 0.55**
LFIS	**505.53 $\pm$ 0.95**	**-111.60 $\pm$ 0.01**	**-108.38 $\pm$ 0.01**	**-109.99 $\pm$ 0.08**
DDS	503.01 $\pm$ 0.77	**-111.58 $\pm$ 0.12**	-108.92 $\pm$ 0.26	**-110.02 $\pm$ 0.06**
PIS	**506.34 $\pm$ 0.63**	-111.69 $\pm$ 0.16	-109.35 $\pm$ 0.74	-109.96 $\pm$ 0.09
AFT	**505.96 $\pm$ 1.19**	-121.63 $\pm$ 16.37	-104.79 $\pm$ 68.31	**-110.07 $\pm$ 0.36**

3.2. Results

Tables 1 and 2 show the performance of different methods on type-1 and type-2 problems respectively. We visualize the samples and weight distributions of type-1 problems in Fig. 2. Comprehensive results of different experimental settings can be found in Appendix D. Our numerical results showed that LFIS is capable of generating competitive samples to existing methods. For two type-1 applications, we observed reasonable estimates of $\log \mathcal{Z}$. Ground-truth (GT) samples can be generated for these two test problems, and the samples generated by LFIS have the lowest sample-to-GT-sample Wasserstein-2 distance. We remark that LFIS performs even better than the conventionally regarded gold-standard SMC with the same number of scales ($T = 256$). For Effective Sample Size (ESS) (Liu, 2008), LFIS also shows competitive performance among all methods studied. Note that both SMC and AFT perform resampling when ESS is lower than a certain threshold (0.98 for SMC and 0.3 for AFT (Arbel et al., 2021a)), while LFIS, DDS, PIS do not resample. Even without resampling, LFIS can still achieve high ESS comparable to SMC with a high resampling threshold. The samples generated by LFIS also provide more uniform coverage over the different modes of the Gaussian mixture and the tails of funnel distribution compared to all other methods. In addition, the weight distributions of LFIS in these two problems are significantly narrower than other methods, explaining the higher ESS and more accurate estimation. For type-2 problems, LFIS is the only method delivering estimates of $\log \mathcal{Z}$ that are consistent with the gold-standard SMC (with $T = 1024$) on all problems.

4. Discussion

We first provide a discussion contrasting LFIS to existing methods and highlight the originality of LFIS. Among many existing methodologies for sampling unnormalized density functions, most closely related to LFIS are sampling by tempered transitions (Neal, 1996), AIS (Neal, 2001), and SMC (Del Moral et al., 2006)), VI-NF (Rezende & Mohamed, 2015), AFTMC (Arbel et al., 2021a), PIS (Zhang & Chen, 2022), CR-ACFMC (Matthews et al., 2022), and DDS (Vargas et al., 2023a). Table 15 provides a summary of the differences between LFIS and other existing methods.

LFIS shares the same spirit of a collection of MC methods (Neal, 1996; 2001; Del Moral et al., 2006), which all used a series of tempered densities for guiding the samplers converging to the target distribution. We adopted the specific form of the time-dependent distribution for type-1 applications from Neal (1996) and Neal (2001), and for type-2 applications from Del Moral et al. (2006). In addition, the weights derived by change-of-measure (see Appendix B) align with the sequential importance sampling technique employed in these MC methods. The distinct difference between our proposed method to these MC-based methods is the transition dynamics between the prescribed series of densities: our proposition is a deterministic flow, in contrast to the stochastic Markov chains (Neal, 1996; 2001) or particle filters (Del Moral et al., 2006). The performance of these MC-based methods critically depends on the choice of transition kernels, which are often an MCMC step, used for evolving samples between consecutive intermediate densities. This requires extensive engineering, i.e., tuning the meta parameters. In contrast, taking a flow-based model to evolve the samples allows us to derive the key equation (5) and formulate an equation-based learning problem, which may streamline the modeling process.

VI-NF (Rezende & Mohamed, 2015) uses a normalizing flow as the modeling distribution, which is parametrized by variational inference. LFIS shares many similar features of VI-NF when a Neural ODE (Chen et al., 2018) is used as the flow. In this case, both our proposition and VI-NF draw samples from simple initial distributions (μ in our case; the base distribution in VI-NF) and use the NN-modeled flow to transport the samples to a future time. Both methods leverage the computable density function along the sample trajectory, a unique feature of flow-based models (see Eq. (8)). The key difference between our proposition and VI-NF is the learning target. VI-NF is an end-to-end approach, that the loss function only depends on matching the

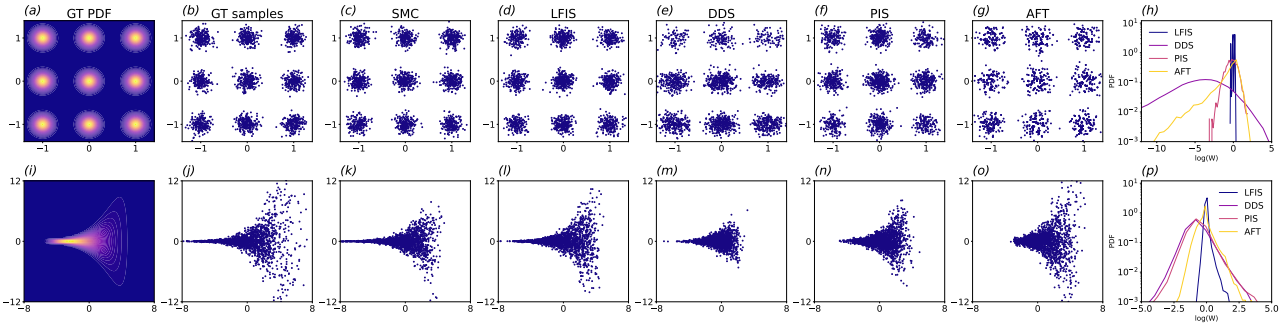

Figure 2. Sampling performance and sample weight distributions for the type-1 problems: (*a-h*) 2-D Gaussian mixture and (*i-p*) 10-D funnel distribution. Subfigures (*a,i*) show the ground-truth PDF contours (marginalized 2-D contour for the funnel distribution). Subfigures (*b-g* , *j-o*) compare the generated samples from the ground-truth distribution and different sampling methods. For the funnel distribution, the samples are projected onto (x_0, x_1) plane. Subfigures (*h,p*) show the log-weight distributions for different sampling methods.

final distribution at the end of the flow to the target distribution. In contrast, LFIS aims to capture $\tilde{\rho}_*(t)$, $t \in [0, 1]$. We hypothesized that providing the whole evolution $\tilde{\rho}_*(t)$, instead of only the final $\tilde{\nu} = \rho_*(1)$, could improve the learning[3]. Moreover, VI-NF aims to directly optimize the reverse Kullback–Leibler (KL) divergence. In contrast, our approach first establishes the PDE (5) for the *optimal flow* which perfectly matches the evolution of the target distribution. Then, we parametrize the neural flow model by minimizing its discrepancy to the evolutionary equation, an approach similar to the Physics Informed Neural Networks (Raissi et al., 2019; Karniadakis et al., 2021). Because VI-NF has been shown to perform worse than AFTMC (Arbel et al., 2021a), we did not include VI-NF in our quantitative comparison in Sec. 3.

The theoretical underpinning of LFIS, Eq. (5), appeared in Vaikuntanathan & Jarzynski (2008) and AFTMC (Arbel et al., 2021a). However, our proposition is significantly different from these studies. In the context of stochastic thermodynamics, Vaikuntanathan & Jarzynski (2008) aimed to obtain the time-dependent density function for a prescribed time-dependent velocity field, contrasting our focus where we provide the unnormalized density function to solve for the velocity field. On the other hand, AFTMC sought to learn normalizing flows but entwined these flows and Monte Carlo kernels across the tempered scales. An ablation study in Appendix E demonstrates that the bulk of learning in AFTMC was predominantly driven by the Monte Carlo kernels, aligning it more closely with SMC (Del Moral et al., 2006) than a pure flow model like LFIS or VI-NF. Without the normalizing flows, AFTMC is functionally identical to SMC. Given that the Monte Carlo kernels can be refined through optimization and that plain SMC competently handles the test problems (as observed in Appendix D.10), the true advantage brought forth by the inclusion of normalizing flows in AFTMC remains nebulous. We remark that CR-AFTMC (Matthews et al., 2022), similar to the construct of AFTMC but has some nuanced differences, also relies on the MC step. Gibbs Flow Approximation (GFA, Heng et al. (2021)) also proposed to leverage the Liouville equation for learning the velocity field, employing a Gibbs sampler instead of relying on Eq. (5) and its associated importance sampling. GFA also integrated Monte Carlo kernels in a manner akin to AFTMC. Nonetheless, it remains uncertain whether the dimension-by-dimension transportation framework of GFA is capable of addressing high-dimensional inference problems effectively.

LFIS is significantly different from other hybrid methods combining AIS and NF, for example, Wu et al. (2020); Geffner & Domke (2021); Zhang et al. (2021); Thin et al. (2021); Doucet et al. (2022); Geffner & Domke (2023). This class of models also rely heavily on Monte Carlo sampling and thus requires special engineering i.e. fine-tuning the Monte Carlo kernel on a case-by-case problem setting.

PIS (Zhang & Chen, 2022) and DDS (Vargas et al., 2023a) are diffusion-based models that share many similar features. PIS learns the drift of an Itô process and DDS learns the time-dependent score function. Both PIS and DDS parametrize the neural network in the path space, by minimizing the reverse KL divergence from a reference process[4] to a modeling process, where the importance sampling is done through a change of measure via the Girsanov theorem (Protter & Protter, 2005). PIS and DDS both are specific examples of a wider class of Schrödinger Bridge problem (Pavon, 1989; Dai Pra, 1991; De Bortoli et al., 2021). Sim-

[3] As the same philosophy applies to AIS and SMC (analogous to LFIS) to simple MCMC-type inference (analogous to VI-NF).

[4] We adopted the nomenclature of DDS, which used "reference process" (PIS used the term "prior uncontrolled process"). A subtle difference between PIS and DDS is the choice of this reference process: PIS uses the standard Wiener process, while DDS uses either over- or under-damped Ornstein–Uhlenbeck process.

ilar to VI-NF, PIS and DDS are both end-to-end and only the terminal distribution ν influences the learning. The performance of this class of models critically depends on the choice of the reference process and how it covers the target distribution. LFIS is closely related to the path-based philosophy of PIS and DDS, but it aims to learn the entire path $\rho_*(t)$, $t \in [0, 1]$. PIS, DDS, and LFIS all estimate the weights of the samples, which can be understood as an integration of the deviation from some optimality along the paths. As a deterministic flow, the path-measure of a particular trajectory with LFIS is always singular (δ distributions) and Girsanov transformation is not applicable. Consequently, the mathematical derivation of the importance sampler for LFIS is original and significantly different from the ones in PIS and DDS. We remark that methods using change of measure as a correction of imperfect computation exist, in addition to AFTMC, PIS, and DDS, see (Chorin & Tu, 2009; Morzfeld et al., 2015; Goodman et al., 2016; Leach et al., 2018). However, it is beyond the authors' knowledge that the accumulated error of Eq. (5) along the trajectory plays a role as the sample weight for a deterministically transported dynamical system. Importance sampling by quantifying the accrued error along the trajectories significantly improved the accuracy of the model (see Appendix D.6). As such, we advocate the novelty of the theoretical construct of LFIS. LFIS is more efficient in sampling than PIS without the need to generate random sample paths, but similar to DDS as it utilizes probability flow ODE (Song et al.), which is a flow model and can be parametrized by our proposed method[5].

LFIS is a sequential importance sampler and it accrues error as the samples evolve. This leads to an expansion in the spread of the weight distribution, consequently reducing the corresponding ESS of the sampler and diminishing its overall quality. This phenomenon is commonly observed in most sequential importance samplers, e.g., AIS, SMC, AFTMC, and CR-AFTMC. To mitigate this issue, one common approach is to perform resampling of the samples as in SMC, AFTMC and CR-AFTMC. In our experiments, we did not observe the weight distribution of LFIS deteriorating to the extent where resampling was deemed necessary. All the results in this manuscript are without resampling, although the incorporation of resampling into LFIS remains a viable option. Conducting LFIS without resampling can be viewed as a stringent test when compared to methods like SMC, AFTMC, or CR-AFTMC. Nonetheless, we still observed narrower weight distributions in LFIS, when compared to other methods, as illustrated in Figure 2 (h) and (p) and also in type-2 problems (data not shown).

We conclude the manuscript by listing the potential limitations of LFIS. (1) LFIS cannot handle non-differentiable density function $\tilde{\nu}(x)$. For example, bounded uniform priors. Theoretically, these singularities can be handled by inserting sources at the discontinuities, a direction that merits future developments. A practical solution to address the challenge is to consider the non-differentiable density function as the limit of differentiable $\tilde{\rho}_*(x, t)$. For example, using sigmoid functions $\tilde{\nu}(x, t) = \sigma(tx/(1-t))$ which converge to step functions $\tilde{\nu}(x) = \Theta(x)$. (2) When the flow satisfying Eq. (5) is too complex, LFIS' approach may require a more expressive NN than end-to-end DDS and PIS. This can result in a more resource-intensive training process, but we speculate that it might offer a tradeoff in terms of improved accuracy. (3) LFIS is more memory-demanding as it requires $\nabla_x S_*$ in the equation, and cross terms like $\partial_\theta \partial_x v_\theta(x, t)$ in the optimization step. These terms are currently evaluated by memory-demanding `autograd`. (4) As an integrator and with the currently adopted explicit scheme, LFIS does not perform well when T is small due to the error induced by a finite time-step (see Fig. 7). This may result in a higher training cost. Optimizing LFIS using higher-order integration schemes and an interpolation of the neural flow along the time domain merits future research.

Code availability. The Feynmann Center for Innovation of Los Alamos National Laboratory is currently reviewing our codes for unlimited release. Once approved, we will deposit the codes at `https://github.com/lanl/LFIS`.

Acknowledgments. The authors acknowledge continual support from Laboratory Directed Research and Development (LDRD). YT was supported by LDRD project "Accelerated Dynamics Across Computational and Physical Scales" (220063DR), NP was supported by LDRD project "Learning Uncertainties In Coupled-Physics Models via Operator Theory" (20230254ER), and YTL was supported by LDRD project "Diffusion Modeling with Physical Constraints for Scientific Data" (20240074ER). YTL sincerely thanks Prof. A. Doucet for several insightful email exchanges that partially inspired this work.

Broader impact. This manuscript presents work whose goal is to advance the field of statistical sampling and machine learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

[5]The drift of the probability flow has a noise-induced term that depends on the score function already, so Eq. (5) for probability flow ODE will have an even higher-order derivative $\nabla_x^2 \log \rho_*(t)$.

References

Arbel, M., Matthews, A., and Doucet, A. Annealed flow transport monte carlo. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 318–330. PMLR, July 2021a.

Arbel, M., Matthews, A., and Doucet, A. Github repository of Annealed Flow Transport Monte Carlo Sampler. https://github.com/google-deepmind/annealed_flow_transport, 2021b. [Online; accessed 4-Jan-2024].

Bhatia, H., Norgard, G., Pascucci, V., and Bremer, P.-T. The Helmholtz-Hodge decomposition—a survey. *IEEE Transactions on visualization and computer graphics*, 19 (8):1386–1404, 2012.

Bierkens, J., Fearnhead, P., and Roberts, G. The Zig-Zag process and super-efficient sampling for Bayesian analysis of big data. *The Annals of Statistics*, 47(3), June 2019. ISSN 0090-5364. doi: 10.1214/18-AOS1715.

Bonneel, N., Rabin, J., Peyré, G., and Pfister, H. Sliced and Radon Wasserstein Barycenters of measures. 51(1):22–45. ISSN 1573-7683. doi: 10.1007/s10851-014-0506-3.

Chen, R. T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Chorin, A. J. and Tu, X. Implicit sampling for particle filters. *Proceedings of the National Academy of Sciences*, 106(41):17249–17254, October 2009. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.0909196106.

Dai Pra, P. A stochastic control approach to reciprocal diffusion processes. *Applied Mathematics and Optimization*, 23(1):313–329, January 1991. ISSN 1432-0606. doi: 10.1007/BF01442404.

De Bortoli, V., Thornton, J., Heng, J., and Doucet, A. Diffusion Schrödinger Bridge with applications to score-based generative modeling. In *Advances in Neural Information Processing Systems*, volume 34, pp. 17695–17709, 2021.

Del Moral, P., Doucet, A., and Jasra, A. Sequential Monte Carlo Samplers. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 68(3):411–436, 2006. ISSN 13697412, 14679868.

Doucet, A., Grathwohl, W., Matthews, A. G., and Strathmann, H. Score-based diffusion meets annealed importance sampling. In *Advances in Neural Information Processing Systems*, volume 35, pp. 21482–21494, 2022.

Duane, S., Kennedy, A., Pendleton, B. J., and Roweth, D. Hybrid Monte Carlo. *Physics Letters B*, 195(2): 216–222, September 1987. ISSN 0370-2693. doi: 10.1016/0370-2693(87)91197-X.

Evans, L. C. *Partial differential equations*, volume 19. American Mathematical Society, 2022.

Faulkner, M. F. and Livingstone, S. Sampling algorithms in statistical physics: A guide for statistics and machine learning, June 2023.

Gabrié, M., Rotskoff, G. M., and Vanden-Eijnden, E. Adaptive Monte Carlo augmented with Normalizing Flows. *Proceedings of the National Academy of Sciences*, 119 (10):e2109420119, March 2022. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.2109420119.

Geffner, T. and Domke, J. MCMC variational inference via uncorrected Hamiltonian annealing. In *Advances in Neural Information Processing Systems*, volume 34, pp. 639–651, 2021.

Geffner, T. and Domke, J. Langevin Diffusion Variational Inference. In *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pp. 576–593. PMLR, April 2023.

Gelman, A. and Meng, X.-L. Simulating normalizing constants: From importance sampling to Bridge Sampling to Path Sampling. *Statistical Science*, 13(2), May 1998. ISSN 0883-4237. doi: 10.1214/ss/1028905934.

Gerlich, G. Die verallgemeinerte Liouville-Gleichung. *Physica*, 69(2):458–466, November 1973. ISSN 0031-8914. doi: 10.1016/0031-8914(73)90083-9.

Geweke, J. Bayesian inference in econometric models using monte carlo integration. *Econometrica: Journal of the Econometric Society*, pp. 1317–1339, 1989.

Goodman, J., Lin, K. K., and Morzfeld, M. Small-Noise Analysis and Symmetrization of Implicit Monte Carlo Samplers. *Communications on Pure and Applied Mathematics*, 69(10):1924–1951, October 2016. ISSN 0010-3640, 1097-0312. doi: 10.1002/cpa.21592.

Green, P. J. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82(4):711–732, 12 1995. ISSN 0006-3444. doi: 10.1093/biomet/82.4.711.

Heng, J., Doucet, A., and Pokern, Y. Gibbs flow for approximate transport with applications to bayesian computation. 83(1):156–187, 2021. ISSN 1369-7412. doi: 10.1111/rssb.12404.

Hénin, J., Lelièvre, T., Shirts, M. R., Valsson, O., and Delemotte, L. Enhanced sampling methods for molecular dynamics simulations [Article v1.0]. *Living Journal of Computational Molecular Science*, 4(1):1583, December 2022. doi: 10.33011/livecoms.4.1.1583.

Hines, K. E. A primer on Bayesian inference for biophysical systems. *Biophysical Journal*, 108(9):2103–2113, 2015. ISSN 0006-3495. doi: 10.1016/j.bpj.2015.03.042.

Jackman, S. Estimation and Inference via Bayesian Simulation: An Introduction to Markov Chain Monte Carlo. *American Journal of Political Science*, 44(2):375, April 2000. ISSN 00925853. doi: 10.2307/2669318.

Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., and Yang, L. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, June 2021. ISSN 2522-5820. doi: 10.1038/s42254-021-00314-5.

Kass, R. E. and Raftery, A. E. Bayes factors. *Journal of the American Statistical Association*, 90(430):773–795, 1995. doi: 10.1080/01621459.1995.10476572.

Larsson, S. and Thomée, V. *Partial differential equations with numerical methods*, volume 45. Springer, 2003.

Leach, A., Lin, K. K., and Morzfeld, M. Symmetrized importance samplers for stochastic differential equations. *Communications in Applied Mathematics and Computational Science*, 13(2):215–241, June 2018. ISSN 2157-5452, 1559-3940. doi: 10.2140/camcos.2018.13.215.

Li, L., Holbrook, A., Shahbaba, B., and Baldi, P. Neural network gradient Hamiltonian Monte Carlo. *Computational Statistics*, 34(1):281–299, March 2019. ISSN 0943-4062, 1613-9658. doi: 10.1007/s00180-018-00861-z.

Lipman, Y., Chen, R. T. Q., Ben-Hamu, H., Nickel, M., and Le, M. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Liu, J. S. *Monte Carlo Strategies in Scientific Computing*. Springer Series in Statistics. Springer, New York, NY, 2. ed edition, 2008. ISBN 978-0-387-76369-9.

Llorente, F., Martino, L., Delgado, D., and López-Santiago, J. Marginal likelihood computation for model selection and hypothesis testing: An extensive review. *SIAM Review*, 65(1):3–58, 2023. doi: 10.1137/20M1310849.

Matthews, A., Arbel, M., Rezende, D. J., and Doucet, A. Continual repeated annealed flow transport Monte Carlo. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 15196–15219. PMLR, July 2022.

McCartan, C. and Imai, K. Sequential Monte Carlo for sampling balanced and compact redistricting plans. *The Annals of Applied Statistics*, 17(4), December 2023. ISSN 1932-6157. doi: 10.1214/23-AOAS1763.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953. ISSN 0021-9606. doi: 10.1063/1.1699114.

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

Morzfeld, M., Tu, X., Wilkening, J., and Chorin, A. Parameter estimation by implicit sampling. *Communications in Applied Mathematics and Computational Science*, 10(2): 205–225, September 2015. ISSN 2157-5452, 1559-3940. doi: 10.2140/camcos.2015.10.205.

Neal, R. M. Sampling from multimodal distributions using tempered transitions. *Statistics and Computing*, 6(4):353–366, December 1996. ISSN 1573-1375. doi: 10.1007/BF00143556.

Neal, R. M. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139, April 2001. ISSN 1573-1375. doi: 10.1023/A:1008923215028.

Neal, R. M. Slice sampling. *The Annals of Statistics*, 31(3): 705–767, June 2003. ISSN 0090-5364, 2168-8966. doi: 10.1214/aos/1056562461.

Neal, R. M. MCMC using Hamiltonian dynamics. *arXiv:1206.1901 [physics, stat]*, June 2012. doi: 10.1201/b10905.

Pajor, A. Estimating the Marginal Likelihood Using the Arithmetic Mean Identity. *Bayesian Analysis*, 12(1):261–287, 2017. ISSN 1936-0975. doi: 10.1214/16-BA1001.

Pavon, M. Stochastic control and nonequilibrium thermodynamical systems. *Applied Mathematics and Optimization*, 19(1):187–202, January 1989. ISSN 1432-0606. doi: 10.1007/BF01448198.

Protter, P. E. and Protter, P. E. *Stochastic differential equations*. Springer, 2005.

Raissi, M., Perdikaris, P., and Karniadakis, G. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. ISSN 0021-9991. doi: 10.1016/j.jcp.2018.10.045.

Resnick, S. *A probability path*. Springer, 2019.

Rezende, D. and Mohamed, S. Variational inference with Normalizing Flows. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1530–1538, Lille, France, 07–09 Jul 2015. PMLR.

Robert, C. P. and Wraith, D. Computational methods for Bayesian model choice. *AIP Conference Proceedings*, 1193(1):251, 2009. doi: 10.1063/1.3275622.

Sharma, S. Markov chain Monte Carlo methods for Bayesian data analysis in astronomy. *Annual Review of Astronomy and Astrophysics*, 55(1):213–259, 2017. doi: 10.1146/annurev-astro-082214-122339.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-Based Generative Modeling through Stochastic Differential Equations. Comment: ICLR 2021 (Oral).

Thin, A., Kotelevskii, N., Doucet, A., Durmus, A., Moulines, E., and Panov, M. Monte Carlo variational auto-encoders. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 10247–10257. PMLR, July 2021.

Tokdar, S. T. and Kass, R. E. Importance sampling: a review. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(1):54–60, 2010.

Tong, A., FATRAS, K., Malkin, N., Huguet, G., Zhang, Y., Rector-Brooks, J., Wolf, G., and Bengio, Y. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=CD9Snc73AW. Expert Certification.

Vaikuntanathan, S. and Jarzynski, C. Escorted free energy simulations: Improving convergence by reducing dissipation. *Phys. Rev. Lett.*, 100:190601, May 2008. doi: 10.1103/PhysRevLett.100.190601.

Vargas, F., Grathwohl, W. S., and Doucet, A. Denoising Diffusion Samplers. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023a.

Vargas, F., Grathwohl, W. S., and Doucet, A. Github repository of Denoising Diffusion Samplers. https://github.com/franciscovargas/denoising_diffusion_samplers, 2023b. [Online; accessed 4-Jan-2024].

Wainwright, M. J. and Jordan, M. I. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1–2):1–305, 2008. ISSN 1935-8237. doi: 10.1561/2200000001.

Wilkinson, D. J. Bayesian methods in bioinformatics and computational systems biology. *Briefings in Bioinformatics*, 8(2):109–116, April 2007. ISSN 1467-5463. doi: 10.1093/bib/bbm007.

Wu, H., Köhler, J., and Noe, F. Stochastic normalizing flows. In *Advances in Neural Information Processing Systems*, volume 33, pp. 5933–5944, 2020.

Zellner, A. Optimal information processing and Bayes's theorem. *The American Statistician*, 42(4):278–280, 1988. doi: 10.1080/00031305.1988.10475585.

Zhang, G., Hsu, K., Li, J., Finn, C., and Grosse, R. B. Differentiable annealed importance sampling and the perils of gradient noise. In *Advances in Neural Information Processing Systems*, volume 34, pp. 19398–19410, 2021.

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

A. Existence of Velocity Field

We prove theorem 2.1. We want to state that uniqueness is not guaranteed, however the existence of the velocity field in Eq. (5) *only depends* on the condition that $\nabla_x \cdot S_* (x,t)$ is bounded for all t *almost everywhere*. Our proof shows that many solutions can exist, i.e., non-uniqueness. Our proposed Liouville Flow Importance Sampler models the velocity field using a neural network that satisfies Eq. (5) in a least squares sense, nonetheless the existence of the velocity field rests on a concrete mathematical foundation based on the Helmholtz–Hodge decomposition as shown below.

Proof. Theorem 2.1. We first show existence. Let $t \in [0,1]$. By Helmholtz–Hodge decomposition (see the survey in (Bhatia et al., 2012)), we can decompose the velocity field $v(x,t)$ as

$$v(x,t) = \nabla_x \psi(x;t) + u(x;t), \tag{22a}$$

where $u(x;t)$ is divergence free, that is $\nabla_x \cdot u = 0$. Plugging this decomposition into Eq. (5), we obtained,

$$\nabla_x^2 \psi(x;t) + S_*(x,t) \cdot \nabla_x \psi(x;t) + S_*(x,t) \cdot u(x;t) = -f(x,t). \tag{22b}$$

Next, by adding and subtracting $c\psi(x;t)$ to the equation above,

$$\nabla_x^2 \psi(x;t) + S_*(x,t) \cdot \nabla_x \psi(x;t) + c\psi(x;t) + S_*(x,t) \cdot u(x;t) - c\psi(x;t) = -f(x,t). \tag{22c}$$

By demanding that ψ and u satisfy the following equations respectively,

$$\nabla_x^2 \psi(x;t) + S_*(x,t) \cdot \nabla_x \psi(x;t) + c\psi(x;t) = -f(x,t); \quad \psi(x,t) = 0 \quad \text{on boundary}, \tag{22d}$$

and

$$S_*(x,t) \cdot u(x;t) = c\psi(x;t), \tag{22e}$$

we see that $v(x,t) = \nabla_x \psi(x;t) + u(x;t)$ satisfies Eq. (5). Observe that the equation in (22d) is an *elliptic partial differential equation* of the form

$$-\nabla_x \cdot (a \nabla_x \psi) + b \cdot \nabla \psi + d\psi = f, \tag{22f}$$

where $a = 1$, $b = -S_*$ and $d = -c$. By the Lax–Milgram theorem (see chp. 6 in (Evans, 2022) for a general description and sec 3.5 in (Larsson & Thomée, 2003) for the particular case described by Eq. (22f)), *a unique* $\psi(x,t)$ *exists* provided $d - \frac{1}{2}\nabla_x \cdot b \geq 0$ for all x, t. In other words, $2d \geq -\nabla_x \cdot S_*(x;t)$, that is $2c \leq \nabla_x \cdot S_*$. By our assumption, this is indeed true.

Observe that if $v_1(x,t)$ is a solution to Eq. (5) and let $\tilde{v}(x,t)$ be a vector field such that $[\nabla + S_*] \cdot \tilde{v} = 0$ then $v_2(x,t) = v_1(x,t) + \tilde{v}(x,t)$ is also a solution to Eq. (5). As $[\nabla + S_*] \cdot \tilde{v} = 0$ only provides one constraint on the D-dimensional field $\tilde{v}$, for $D \geq 2$, there is no unique solution to Eq. (5).

□

B. Importance Sampling for Imperfect Neural Networks

Almost surely, a finite network cannot accurately model the velocity field that satisfies Eq. (5). Let us denote the neural velocity field by $v_\theta(x,t)$, where θ stands for the weights of the neural network. In this scenario, we denote the induced density function by $\rho_\theta(x,t)$, which in general is not the target density function $\rho_*(x,t)$. Note that GLE (1) ensures that the trajectories of N samples following the deterministic evolution (2) as given by Eq. (7) share the same statistics of $\rho_\theta(\cdot,t)$. That is, $x^{(i)}(t) \sim \rho_\theta(\cdot,t)$ whenever $x_0^{(i)} \sim \mu$. Our goal is to derive an equation that is analogous to Eq. (5) using variational inference (VI) for this imperfect ($\rho_\theta \neq \rho_*$) scenario.

To describe an instantaneous equation for the velocity field, it suffices to consider performing variational inference at time $t + \mathrm{d}t$, where the infinitesimal advanced time $\mathrm{d}t \ll 1$. Using VI, we aim to minimize the reverse Kullback–Leibler divergence $\mathrm{KL}\left(\rho_\theta\left(\cdot, t+\mathrm{d}t\right) \| \rho_*\left(\cdot, t+\mathrm{d}t\right)\right)$. As we will be using the sample trajectories, it is convenient to first quantify both $\log \rho_\theta(x(t),t)$ and $\log \rho_*(x(t),t)$ along the trajectory. Note that we are adopting the Lagrangian specification of

the flow because we are computing the quantity of interests along the trajectory. Below, for brevity, we denote a sample trajectory $x^{(i)}(t)$ by the abbreviated $x(t)$. We first compute the instantaneous rate of log-densities along the trajectories:

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\log\rho_\theta(x(t),t) &= \partial_x\left[\log\rho_\theta(x(t),t)\right]\cdot\frac{\mathrm{d}x(t)}{\mathrm{d}t}+\partial_t\log\rho_\theta(x(t),t)\\ &= S_\theta(x(t),t)\cdot v_\theta(x(t),t)-\frac{1}{\rho(x(t),t)}\nabla_x\cdot\left(v_\theta(x(t),t)\rho_\theta(x(t),t)\right) \qquad \text{(By Eq. (1))}\\ &= -\nabla_x\cdot v_\theta\left(x(t),t\right), \qquad (23)\\ \frac{\mathrm{d}}{\mathrm{d}t}\log\rho_*(x(t),t) &= \partial_x\left[\log\rho_*(x(t),t)\right]\cdot\frac{\mathrm{d}x(t)}{\mathrm{d}t}+\partial_t\log\rho_*(x(t),t)\\ &= S_*(x(t),t)\cdot v_\theta(x(t),t)+\partial_t\delta\log\tilde{\rho}_*(x(t),t), \qquad (24)\end{aligned}$$

where we have used $\rho_*(x,t)=\tilde{\rho}_*(x,t)/\mathcal{Z}(t)$ and Eq. (5b). From the above equations, it is clear that the log-densities along the trajectories can be computed along the dynamics via

$$\log\rho_\theta(x(t),t) = \log\rho_\theta(x(0),0)-\int_0^t\nabla\cdot v_\theta\left(x(s),s\right)\,\mathrm{d}s, \qquad (25)$$

$$\log\rho_*(x(t),t) = \log\rho_*(x(0),0)+\int_0^t\left[S_*(x(s),s)\cdot v_\theta(x(s),s)+\partial_t\delta\log\tilde{\rho}_*(x(s),s)\right]\mathrm{d}s \qquad (26)$$

We remark that Eq. (25) is the pivotal construct to enable density estimation by Neural ODE (Chen et al., 2018). Because the initial distribution $\rho_\theta(\cdot,0)=\mu(0)=\rho_*(\cdot,0)$ we get the following important result,

With ρ_θ and ρ_* defined as above, we have the following density ratio

$$\log\frac{\rho_*(x(t),t)}{\rho_\theta(x(t),t)} = \int_0^t\left[\nabla\cdot v_\theta\left(x(s),s\right)+S_*(x(s),s)\cdot v_\theta(x(s),s)+\partial_t\delta\log\tilde{\rho}_*(x(s),s)\right]\mathrm{d}s. \qquad (27)$$

The above equation hints that we should define a dynamic error function along a specific sample trajectory

$$\varepsilon(t;x_0)\equiv\nabla\cdot v_\theta\left(x(t;x_0),t\right)+S_*(x(t;x_0),t)\cdot v_\theta(x(t;x_0),t)+\partial_t\delta\log\tilde{\rho}_*\left(x(t;x_0),t\right), \qquad (28)$$

which is identically the difference between the LHS and RHS of Eq. (5) along the trajectory. We remark that the error function defined in (6) is consistent to the definition (28) above, but we have dropped the time-dependence in Eq. (6). Then, for every trajectory $x(t)$, we associate a dynamic weight $w(t;x_0)$ to the trajectory

$$\tilde{w}(t;x_0):=\frac{\rho_*(x(t;x_0),t)}{\rho_\theta(x(t;x_0),t)}=\exp\left(\int_0^t\varepsilon(s;x_0)\,\mathrm{d}s.\right) \qquad (29)$$

Note that because $x(t;x_0)$ depends on the initial condition x_0, the dynamic error function $\varepsilon(t;x_0)$ and the difference between the log-densities also depend on the initially sampled $x_0\sim\mu$.

Proposition B.1. *Assuming ρ_θ dominates ρ_*. In the optimal case when $\varepsilon(x,t)=0\ \forall x$ in the support of ρ_θ, the density function ρ_θ induced by the (trained) neural-network-modeled flow is identical to that of the target distribution, ρ_*. In this case, $\tilde{w}=1$.*

Proof. By Eqs. (27) and (29). □

Equation (29) plays a key role in the following analysis.

Theorem B.2. *Given a neural velocity field $v_\theta(x,t)$ and an integrable function $F(x)$. Then for any $t\geq 0$,*

$$\mathbb{E}_{x\sim\rho_\theta(\cdot,t)}\left[F(x)\right]=\mathbb{E}_{x_0\sim\mu}\left[F\left(x\left(t;x_0\right)\right)\right], \qquad (30)$$

where $\rho_\theta\left(\cdot,t\right)$ is the solution of the following Generalized Liouville Equation

$$\partial_t\rho_\theta\left(x,t\right)=-\nabla_x\cdot\left[v_\theta\left(x,t\right)\rho_\theta\left(x,t\right)\right] \qquad (31)$$

with initial data

$$\rho_\theta\left(x,0\right)=\mu(x). \qquad (32)$$

Proof. This Lemma follows directly from the definition of $\rho_\theta(\cdot, t)$ in the Generalized Liouville Equation, that $\rho\,(x, t)$ is the density of the trajectory $x(t)$ with an initial condition $x_0 \sim \mu$. See Gerlich (1973). □

Remark B.3. Lemma B.2 is the theoretical foundation of Neural ODE (Chen et al., 2018), a continuous-time normalizing flow.

Next, we show that the ratio of the log-densities in Eq. (29) provides a way for us to estimate the expectation of a measurable function F with respect to the target density $\rho_*\,(\cdot, t)$ by a change-of-measure, without the need of generating samples from the target distribution.

Lemma B.4. *Assuming ρ_θ dominates ρ_*, for any $t \in [0, 1]$,*

$$\mathbb{E}_{x' \sim \rho_*(\cdot,t)}\left[F(x')\right] = \mathbb{E}_{x_0 \sim \mu}\left[\tilde{w}(t; x_0)\, F\left(x\left(t; x_0\right)\right)\right]. \tag{33}$$

Proof.

$$\begin{aligned}
\mathbb{E}_{x' \sim \rho_*(\cdot,t)}\left[F\left(x'\right)\right] &= \int F\left(x'\right) \rho_*(x', t)\, \mathrm{d}x' \\
&= \int F\left(x'\right) \frac{\rho_*(x', t)}{\rho_\theta(x', t)} \rho_\theta(x', t)\, \mathrm{d}x' \\
&= \mathbb{E}_{x' \sim \rho_\theta(\cdot,t)}\left[F\left(x'\right) \frac{\rho_*(x', t)}{\rho_\theta(x', t)}\right] \\
&= \mathbb{E}_{x_0 \sim \mu}\left[F\left(x\left(t; x_0\right)\right) \frac{\rho_*(x\left(t; x_0\right), t)}{\rho_\theta(x\left(t; x_0\right), t)}\right] \\
&= \mathbb{E}_{x_0 \sim \mu}\left[\tilde{w}\left(t; x_0\right)\; F\left(x\left(t; x_0\right)\right)\right],
\end{aligned} \tag{34}$$

where we used Lemma B.2 to establish the second-to-last equality and the definition Eq. (29) to establish the last equality. □

Remark B.5. We put a stringent condition that ρ_θ has to dominate ρ_* for any arbitrary function F. The condition is often relaxed to "$\rho_\theta(x) > 0$ where $F(x)\rho_*(x) > 0$ in the importance sampling literature (e.g., Tokdar & Kass (2010)), which requires knowledge of the function F.

Lemma B.6. *Assuming ρ_θ dominates ρ_* and $\partial_t \log \tilde{\rho}_*(x, t)$ is integrable with respect to $\rho_*(\cdot, t)$ for any $t \in [0, 1]$,*

$$\mathbb{E}_{x' \sim \rho_*(\cdot,t)}\left[\partial_t \log \tilde{\rho}_*(x', t)\right] = \mathbb{E}_{x_0 \sim \mu}\left[\tilde{w}(t; x_0)\; \partial_t \log \tilde{\rho}_*\left(x(t; x_0), t\right)\right]. \tag{35}$$

Proof. By Lemma B.4 with $F(x) := \partial_t \log \tilde{\rho}_*(x, t)$. □

Proposition B.7. *Given an integrable function F, $\mathbb{E}_{x' \sim \rho_*(\cdot,t)}\left[F\left(x'\right)\right]$ can be unbiasedly estimated by a finite set of initial samples $x_0^{(i)}$,*

$$\mathbb{E}_{x' \sim \rho_*(\cdot,t)}\left[F\left(x'\right)\right] \xrightarrow{\text{i.p.}} F_N(t) := \sum_{i=1}^{N} w_i(t)\, F\left(x\left(t; x_0^{(i)}\right)\right), \tag{36}$$

where normalized weights $w_i(t)$ are defined as (Eq. (10)):

$$w_i(t) := \frac{\tilde{w}\left(t, x_0^{(i)}\right)}{\sum_{j=1}^{N} \tilde{w}\left(t, x_0^{(j)}\right)}. \tag{37}$$

Remark B.8. Note that the convergence result using the weight normalization (Eq. (10)) rests on the following fact

$$\lim_{N \to \infty} \frac{1}{N} \sum_{i=1}^{N} \tilde{w}(t; x_0^{(i)}) F(x(t; x_0^{(i)})) = \lim_{N \to \infty} \sum_{i=1}^{N} w_i(t)\, F\left(x\left(t; x_0^{(i)}\right)\right)$$

which follows from importance sampling. See (Geweke, 1989) for the proof and (Tokdar & Kass, 2010) for a review.

Remark B.9. Note that in practice we get the stronger almost sure convergence in the result above by demanding that $\mathbb{E}_{x_0\sim\mu}|\,\tilde{w}\,(t;x_0)\;F\,(x\,(t;x_0))\,|<\infty$. This is not an unreasonable assumption as each weight $\tilde{w}$ is absolutely bounded when using a suitably expressive neural network.

Corollary B.10. *Assuming $\partial_t \log\tilde{\rho}_*\,(x,t)$ is integrable, $\mathbb{E}_{x'\sim\rho_*(\cdot,t)}\left[\partial_t \log\tilde{\rho}_*\,(x,t)\right]$ can be unbiasedly asymptotically (i.e. unbiased as $N\to\infty$) estimated by a finite set of initial samples $x_0^{(i)}$,*

$$\mathbb{E}_{x'\sim\rho_*(\cdot,t)}\left[F\,(x')\right]\approx\sum_{i=1}^{N} w_i(t)\,\partial_t \log\tilde{\rho}_*\left(x\left(t;x_0^{(i)},t\right)\right). \tag{38}$$

Corollary B.11. *The last term in Eq. (28), $\partial_t\delta\log\tilde{\rho}_*\,(x(t),t)$, can be estimated at time t.*

Proof. By Eq. (5b), $\partial_t\delta\log\tilde{\rho}_*\,(x(t),t)$ contains two terms: $\log\tilde{\rho}_*\left(x^{(i)}\,(t)\,,t\right)$, and $\mathbb{E}_{x'\sim\rho_*(\cdot,t)}\left[\partial_t\log\tilde{\rho}_*(x',t)\right]$. The former can be evaluated straightforwardly as the density function is given. The latter can be estimated by Corollary B.10. □

C. (Asymptotically) Unbiased Estimators of the Marginalized Likelihood

Proof. Theorem 2.2

First, we show that that $\log\mathcal{Z}(t)$ is given by integrating $I(s)$ on $[0,t]$ where $I(s)=\mathbb{E}_{\rho_*(\cdot,s)}\left[\partial_s\log\tilde{\rho}_*\,(x,s)\right]$ i.e.

$$\log\mathcal{Z}(t)=\int_0^t \mathbb{E}_{\rho_*(\cdot,s)}\left[\partial_s\log\tilde{\rho}_*\,(x,s)\right]\,\mathrm{d}s. \tag{39}$$

This can be established by first applying $\mathrm{d}/\mathrm{d}t$ to $\log\mathcal{Z}\,(t):=\int\tilde{\rho}_*\,(x,t)\,\mathrm{d}x$:

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\log\mathcal{Z}(t)&=\frac{\mathrm{d}}{\mathrm{d}t}\log\int\tilde{\rho}_*\,(x,t)\;\mathrm{d}x=\frac{1}{\mathcal{Z}(t)}\int\frac{1}{\tilde{\rho}_*\,(x,t)}\left(\frac{\partial\tilde{\rho}_*\,(x,t)}{\partial t}\right)\tilde{\rho}_*\,(x,t)\;\mathrm{d}x\\&=\int\frac{\partial\log\tilde{\rho}_*\,(x,t)}{\partial t}\frac{\tilde{\rho}_*\,(x,t)}{\mathcal{Z}(t)}\,\mathrm{d}x=\mathbb{E}_{x\sim\rho_*(\cdot,t)}\left[\partial_t\log\tilde{\rho}_*\,(x,t)\right].\end{aligned} \tag{40}$$

Integrating the above equation from 0 to t, we obtained

$$\log\mathcal{Z}(t)=\int_0^t \mathbb{E}_{\rho_*(\cdot,s)}\left[\partial_s\log\tilde{\rho}_*\,(x,s)\right]\,\mathrm{d}s, \tag{41}$$

because $\log\mathcal{Z}(0)=0$ a we fix $\rho_*(t=0)=\mu$, a normalized distribution. We remark that this derivation is akin to that in (Gelman & Meng, 1998). Corollary B.10 provides us with an unbiased estimate of $I(s)$ using normalized weights as in Eq. (10). Thus we have,

$$\lim_{N\to\infty}\sum_{i=1}^{N} w_i\,(t)\;\partial_t\log\tilde{\rho}\left(x^{(i)}\,(t)\,,t\right)\xrightarrow{\text{i.p.}}I(s), \tag{42}$$

for each $s\in[0,t]$. In practice as remark B.9 suggests, we do get stronger almost sure convergence. However, given the assumption that both the dynamic weights $w_i(t)$ and $\partial_t\log\tilde{\rho}_*\left(x^{(i)}(t),t\right)$ are absolutely bounded almost everywhere along the sample trajectories our final result still holds. With this assumption in mind, we observe that the convergence of the integrand in probability

$$\lim_{N\to\infty}\sum_{i=1}^{N} w_i\,(t)\;\partial_t\log\tilde{\rho}\left(x^{(i)}\,(t)\,,t\right)\xrightarrow{\text{i.p.}}\mathbb{E}_{\rho_*(\cdot,s)}\left[\partial_s\log\tilde{\rho}_*\,(\cdot,s)\right] \tag{43}$$

implies the almost-surely convergence of the integration via the Lebesgue Dominated Convergence Theorem (see Corollary 6.3.2 in (Resnick, 2019)). Thus, we have

$$\lim_{N\to\infty}\int_0^t\sum_{i=1}^{N} w_i\,(s)\;\partial_s\log\tilde{\rho}\left(x^{(i)}\,(s)\,,s\right)ds\xrightarrow{\text{a.s.}}\int_0^t\mathbb{E}_{\rho_*(\cdot,s)}\left[\partial_s\log\tilde{\rho}_*\,(\cdot,s)\right]\mathrm{d}s\overset{\text{Eq. (41)}}{=}\log\mathcal{Z}\,(t)\,. \tag{44}$$

□

Remark C.1. The boundedness assumption in Theorem 2.2 can be relaxed to require a dominating integrable function. However, in practice we do see the assumptions met especially if we have a suitably expressive neural velocity model.

We now show the proof of the unbiased estimation of $\mathcal{Z}$, Theorem 2.3.

Proof. Unbiased estimator of $\mathcal{Z}$. We first relate ε defined in (6b) and ϵ in (14):

$$\epsilon\left(x\left(s\right),\theta\right)=\varepsilon\left(x\left(s\right),\theta\right)+\partial_s\left\langle\log\tilde{\rho}_*(\cdot,s)\right\rangle_*, \tag{45}$$

because of the definition Eq. (5b). Integrating both sides from $s=0\rightarrow t$ and using Eqs. (15) (28), (29), and (41), we obtain

$$\log\varpi(t;x_0)=\log w(t;x_0)+\log\mathcal{Z}(t). \tag{46}$$

Exponentiating the above expression leads to

$$\varpi(t;x_0)=w(t;x_0)\,\mathcal{Z}(t). \tag{47}$$

Now, we perform expectation over the empirical distribution ρ_θ and using Eq. (29):

$$\mathbb{E}_{\rho_\theta}\left[\varpi(t;x_0)\right]=\mathbb{E}_{\rho_\theta}\left[w(t;x_0)\,\mathcal{Z}(t)\right]=\mathcal{Z}(t)\,\mathbb{E}_{\rho_\theta}\left[w(t;x_0)\right]=\mathcal{Z}(t)\,\mathbb{E}_{\rho_*}\left[t\right]=\mathcal{Z}(t). \tag{48}$$

We can now unbiasedly estimate $\mathcal{Z}(t)$ using Monte Carlo, i.e., Eq. (16). □

Remark C.2. Our results are in continuous time and the Generalized Liouville Equation requires that we have the exact velocity field. In practice, we use a time integrator to evolve the velocity field in discrete time in chunks of δt. In this paper we use (Forward) Euler time stepping. Hence the results above do not hold in general; in fact there is a bias term which depends on δt. However, in the limit as $\delta t\rightarrow 0$ we do get the unbiased estimate (assuming that our neural network modeled velocity is Lipschitz and not chaotically dependent on the initial condition).

D. Additional Experiment Details and Results

In this section, we provide additional details of the numerical experiments for reproducibility.

D.1. Additional details of the test distributions

For the Mode-separated Gaussian mixture problems, we followed the testing problem proposed in Zhang & Chen (2022) and chose a small variance ($\sigma=0.012$) for each mode of the Gaussian mixture to make it more challenging for different methods. We reduce the grid size to $\{-1,0,1\}^2$ such that using the simplest prior/reference distribution $\mathcal{N}(0,\mathbf{I})$ for different methods could provide good coverage of the target distribution. This testing problem could provide a fair comparison between different methods for samples from well-separated modes in a distribution. Our results show that the proposed LFIS excels at generating evenly distributed samples from different modes of the target distributions.

For the Log Gaussian Cox process, the covariance matrix of the prior distribution K takes the form $K(u,v)=\sigma^2\exp\left(-\frac{|u-v|_2}{M\beta}\right)$, where $\sigma^2=1.91$, $M=40$ is the grid number in each direction, and $\beta=1/33$. Here, u and v denote the normalized positions on two grid points. The mean vector of the prior distribution is $\log(126)-\sigma^2$.

For the sampling problem in the latent space of VAE, we took to decoder $p_\theta(x|z)$ of the pre-trained VAE in Arbel et al. (2021a) and used it to construct the likelihood function with binary cross-entropy loss. In Fig. 3, we demonstrate the test image and the reconstructed image using the pre-trained decoder $p_\theta(x|z)$ from samples generated by LFIS.

D.2. LFIS for multi-modal distributions with unequally weighted modes

In this section, we test LFIS on distributions with unequally weighted modes. We reformulate the mode-separated Gaussian mixture distribution and give each mode different weights. We tested three different configurations of unequally weighted modes whose PDFs are shown in Figs. 4 (a,d,g). The numerical experiments are performed using the exact setup as the regular Gaussian mixture distribution. The samples drawn from the ground-truth distributions and LFIS are shown Fig. 4 (b,c,e,f,h,i). The $\log\mathcal{Z}$ estimation for each configuration is shown in table 3. Evidently, LFIS is capable of sampling from multi-modal distributions with unequally weighted modes.

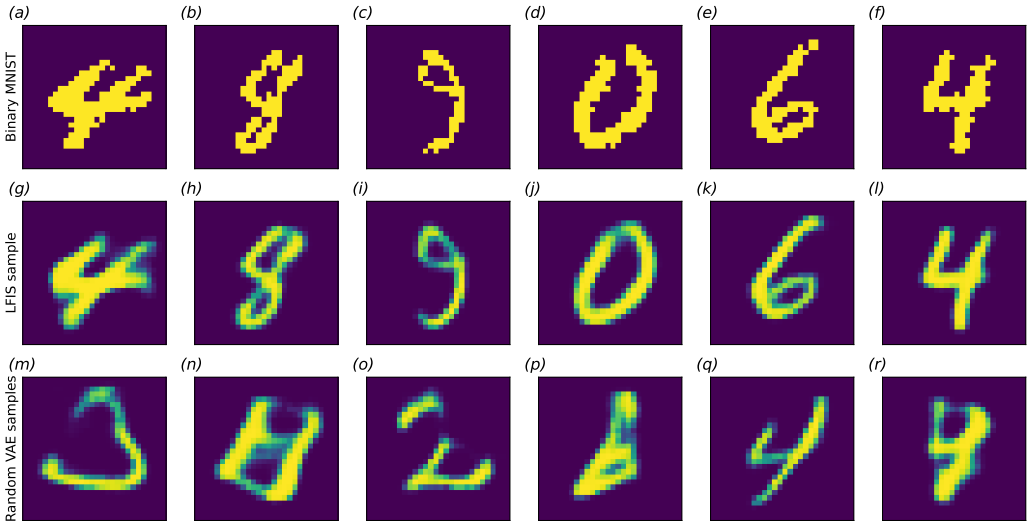

Figure 3. Binary MNIST dataset (*a-f*), decoded LFIS samples (*g-l*), and decoded image from random latent space samples (*m-r*).

Table 3. $\log \hat{\mathcal{Z}}$ estimation for mode-separated Gaussian mixture with unequal weights.

	MG ($D = 2$)
Configuration 1	-0.0016 $\pm$ 0.005
Configuration 2	0.0013 $\pm$ 0.004
Configuration 3	-0.0003 $\pm$ 0.006

D.3. Neural network architecture

The expressibility of the neural network could significantly impact the sampling quality of the LFIS, considering that LFIS imposes a specific density flow from $t = 0 \to 1$ to satisfy $\rho_*(x, t)$, while other methods like PIS, DDS do not enforce this condition. However, in the numerical experiments, we find that an NN similar to those used in PIS and DDS is expressive enough for all the testing problems. For LFIS, we use an NN with 2 hidden layers, each with 64 nodes to parameterize the discrete-time velocity field. For both DDS and PIS, the NNs have time-encoding and are augmented by the gradient information or the score $S(x, t)$.

It is difficult to make a direct comparison of NN structures and numbers of parameters between the discrete-time and continuous-time models. For LFIS, the total number of parameters will increase with the total number of steps. For DDS and PIS, the number of parameters will increase with the number of channels used for time-encoding increasing. Overall, when LFIS uses the same number of time steps as the time-encoding in DDS and PIS, the total number of NN parameters are very similar, thus causing no additional memory footprint.

D.4. Additional details of the training procedure

The general training procedure is summarized in Algorithm 1. For each discretized time step k, at each training epoch, a new minibatch (B) of samples is generated by calling the `GenerateSamples` function. This step is crucial to provide good coverage of the sampling space. However, for low-dimensional problems, repeatedly calling the `GenerateSamples` function for each minibatch could slow down the training process, because the initial samples need to be passed through all the previously trained flow $v_{\theta_*^{(l)}}$. To accelerate the training of low-dimensional sampling problems ($D < 10$), we skip the generation of new samples at each epoch. Instead, we use a subset of the N samples used for estimating $\langle \partial_t \log \tilde{\rho}_* \rangle_*$ for training. When N is large enough to provide good coverage of the sampling space, this simplified training procedure could provide a noticeable acceleration of learning the Liouville flow (add numbers).

The choice of stopping criteria for training the Liouville flow at each time step $v_{\theta_*^{(l)}}$ is also crucial for im-

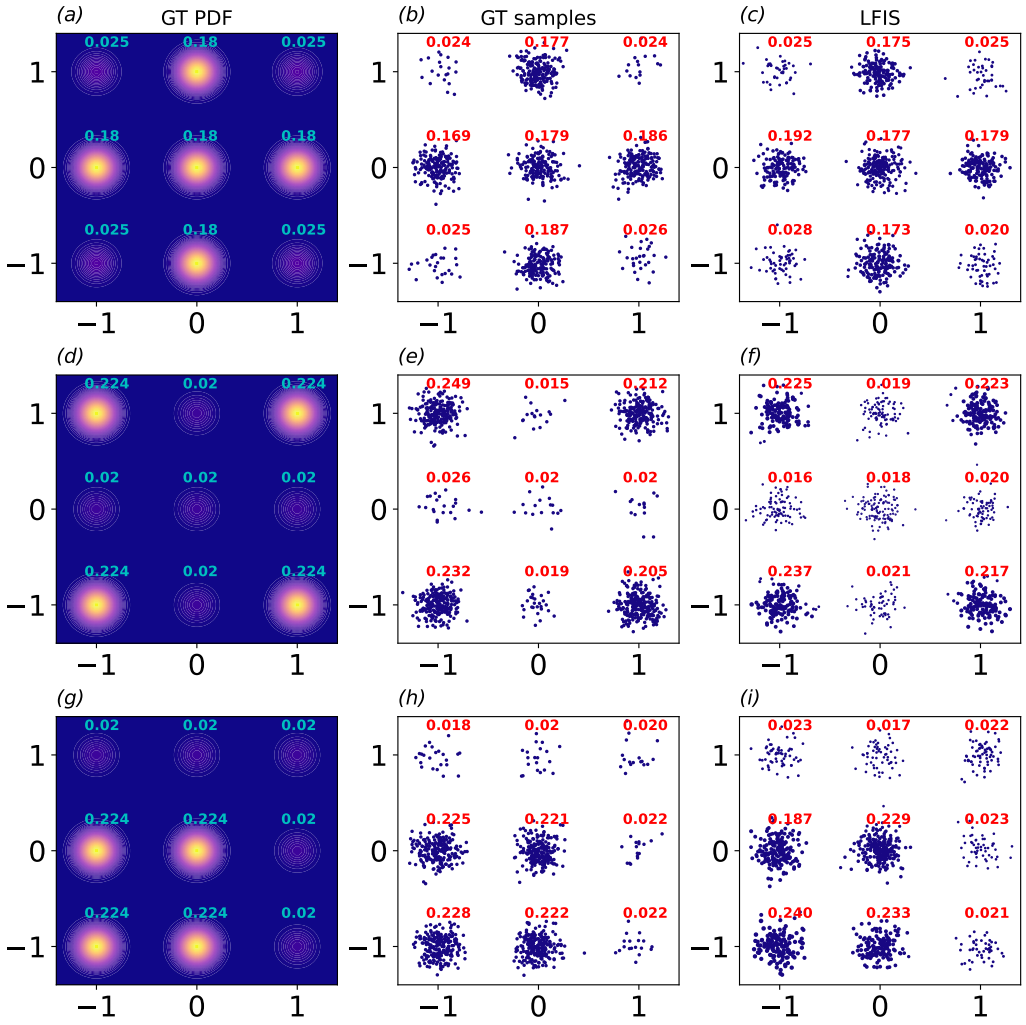

Figure 4. Mode-separated Gaussian mixture with mixed weights for each mode. Three different configures of the mixed weights are considered. Subfigures (*a,d,g*) show the ground-truth PDFs with the corresponding weights annotated by each mode. The samples generated from the ground-truth PDF and LFIS are shown in subfigures (*b,e,h*) and (*c,f,i*), with the sizes of the samples scaled by the sample weights (note that the sample weights are different from the modal wights).

proving the training quality and speed. Through a series of numerical experiments, we introduce the criteria $\mathbb{E}_i\left[\varepsilon^2(x^{(i)}(t);\theta)\right]/\mathrm{var}_i\left[\partial_t \log \tilde{\rho}_*\left(x^{(i)}(t),t\right)\right]$ at each training time, which represents the percentage of the squared error to the total variation of the RHS of the equation. In general, we found that when the criteria converge to less than 0.001 during training, the learned velocity field can provide a good approximation of the Liouville flow. If the proposed criteria can not be met during training, either due to the limited expressibility of the NN, or accumulated error from the previous velocity field, we will stop the training at 2,000 epochs. Empirically, we found the NNs converged to the desired 0.001 for at least half of the discrete timesteps, for most of the test problems. The imperfect representation of the Liouville flow using NN will be quantified using the sample weights during sampling.

For all the numerical experiments, we use the Adam optimizer with an initial learning rate of 5×10^{-3}. We employ an optimizer schedule that will reduce the learning rate to 50% every 200 epochs without observing any improvement in the loss.

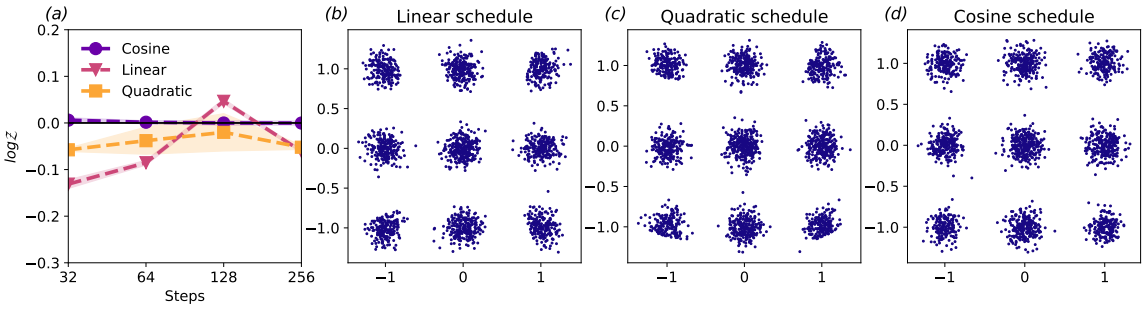

Figure 5. The effects of schedule on $\log \hat{\mathcal{Z}}$ estimation at different Ts and sample quality.

Table 4. The effects of schedule on $\log \hat{\mathcal{Z}}$ estimation.

Schedule	MG ($D = 2$)
Linear	-0.06 $\pm$ 0.003
Quadratic	-0.05 $\pm$ 0.005
Cosine	-0.0002 $\pm$ 0.004

D.5. Choice of schedule

We tested three different choices of the schedule function $\tau(t)$, which controls the "pace" of the annealing: (1) a linear schedule $\tau(t) = t$ (with an even "pace"), (2) a quadratic schedule $\tau(t) = t$ (a slow pace at the beginning when $\rho_*(t)$ is close to μ and fast in the end when $\rho_*(t)$ is close to ν), and (3) a cosine schedule $\tau(t) = (1 - \cos(\pi t))/2$ (slow pace at the beginning when $\rho_*(t)$ is close to μ and in the end when $\rho_*(t)$ is close to ν and fast during intermediate times.) The numerical experiment results are shown in Table 4 and Fig. 5 following the same procedure as described above for estimating $\log \mathcal{Z}$ for the 2D Gaussian mixture problems. As we can see in Fig. 5 (a), the cosine schedule shows better convergence of $\log \mathcal{Z}$ as we increase the number of total time steps. In Fig. 5 (b-d), we show the samples generated using three types of schedules. We observe that the samples from both the linear schedule and quadratic schedule show non-uniform coverage at the edge of the distribution, while the samples generated with the cosine schedule uniformly span over each mode of the Gaussian mixture distribution. Based on these tests, we use the cosine schedule for the rest of the numerical experiments for LFIS.

D.6. Improvement of statistical estimation by sample weights

A novel contribution of LFIS is using the accumulated error induced by imperfectly trained NN as sample weights for unbiased and consistent estimation of statistical quantities. In this section, we provide numerical results showing the improvement of statistics estimation using accumulated errors as weights, using the MG and Funnel test problems which are analytical. In the proposed LFIS algorithm, the weights are used in both training and sampling procedures for computing $\langle \partial_t \log \tilde{\rho}_* \rangle_*$. To measure the effects of sample weights on training and sampling separately, we performed four different combinations of numerical experiments for both MG and funnel distributions. We compare the $\log \mathcal{Z}$ estimation in Table 5. There are no significant improvements in adding sample weights during training. However, by using the accumulated errors as sample weights during sampling, the estimations can be significantly improved in these two models. The results evidentiate the effectiveness of our proposed novel corrections for sampling.

D.7. Training and sampling cost

Similar to PIS and DDS, the proposed LFIS is an NN-parameterized model that can be trained once and then deployed for sampling, while the MC-based algorithms need to be repeated every time they are used to perform sampling. In this section, we focus on comparing training costs and sampling costs within similar type of algorithms that are based on NN.

For LFIS, DDS, and PIS, we performed 30 independent training and sampling processes for the 10D funnel distributions. We set the total steps to be $T = 256$ and trained the NNs of different methods for a maximum of 2,000 epochs or until the

Table 5. $\log \hat{\mathcal{Z}}$ estimation of LFIS with/without weights

		Training w/ weights		Training w/o weights	
	T	Sampling w/ weights	Sampling w/o weights	Sampling w/ weights	Sampling w/o weights
MG ($D = 2$)	32	0.0060 ± 0.005	-0.130 ± 0.050	0.0048 ± 0.005	-0.150 ± 0.061
	64	0.0015 ± 0.003	-0.076 ± 0.051	0.0009 ± 0.004	-0.058 ± 0.045
	128	-0.0001 ± 0.004	-0.032 ± 0.066	0.0001 ± 0.004	-0.042 ± 0.064
	256	-0.0002 ± 0.004	-0.035 ± 0.070	0.0002 ± 0.004	-0.028 ± 0.044
Funnel (D=10)	32	-0.29 ± 0.022	-0.45 ± 0.052	-0.56 ± 0.002	-0.61 ± 0.044
	64	-0.159 ± 0.028	-0.31 ± 0.053	-0.16 ± 0.009	-0.29 ± 0.043
	128	-0.19 ± 0.01	-0.27 ± 0.070	-0.15 ± 0.010	-0.24 ± 0.048
	256	-0.07 ± 0.003	-0.15 ± 0.045	-0.06 ± 0.018	-0.15 ± 0.065

Table 6. Traning and sampling cost

Method	Training time (minutes)	Sampling time (ms)
LFIS (w/o weight, not compiled)	27.6 ± 1.84	90.4 ± 4.18
LFIS (w/ weight, not compiled)	-	526.9 ± 7.48
PIS	36.4 ± 1.51	819.9 ± 3.99
DDS (w/o weight, JIT)	37.13 ± 0.28	51.91 ± 2.27
DDS (w/ weight, JIT)	-	57.10 ± 6.09

convergence criteria were met for each of the methods. The batch size is chosen to be 10,000. The optimizer, learning rate, and learning rate schedule are set to be the same as provided by the original papers. For the sampling experiments, we took the pre-trained models and deployed them to independently generate 30 batches of samples, each with 2,000 points. All the experiments are performed using a single NVIDIA A100 GPU with 40GB of RAM.

Table 6 compares the training and sampling time of different NN-based sampling methods. For training, the LFIS is faster than both PIS and DDS, which is partially due to the proposed stop criteria for the convergence of NN at each discrete time step. On the other hand, DDS has the fastest sampling time because of the probability flow ODE and the just-in-time (JIT) compilation with JAX. However, the first run/compilation of the DDS took 4942ms (with weight computation) and 1528ms (w/o weight computation). As such, DDS with JIT has a worse performance unless one desires multiple sampling with the same trained model. For LFIS, the majority of the sampling time is attributed to the computation of sample weights. In our implementation of LFIS based on PyTorch, the divergence of the velocity field is computed via the `torch.autograd.functional.jacobian` function, which is not optimized for divergence computation, nor compilable currently.

D.8. Reproducibility of LFIS training

We demonstrate the reproducibility of the LFIS sampling statistics by performing the 30 independent trainings from different initializations of the NN parameters. For each of the independently trained LFIS, we run 30 sampling procedures of 2,000 samples to estimate the mean and variance of $\log \mathcal{Z}$. In Fig. 6, we plot the histograms of $\log \mathcal{Z}$ statistics for 30 LFISs. We observe a very narrow spread of $\log \mathcal{Z}$ statistics over all 30 trainings, which shows the reproducibility of the proposed methods. In the main text, we report the statistics that is the closest to the mean of 30 LFISs.

D.9. Choice of total time steps and tempered scales

Here, we provide the results of $T \in \{32, 64, 128, 256\}$ for SMC (Table 7), AFTMC (Table 8), PIS (Table 9), DDS (Table 10), LFIS (Table 11), and a summary plot (Fig. 7). To enforce uniformity, we ran each method 30 times independently and computed the summary statistics from the collected results.

For the two transformation problems: Gaussian mixture and funnel distributions, we perform the numerical experiments under the assumption that the coverage of the distribution is unknown so that the prior/reference distribution is set to be the simplest uncorrelated normal distributions with zero mean and identity covariance matrix. For LFIS, this can be achieved by

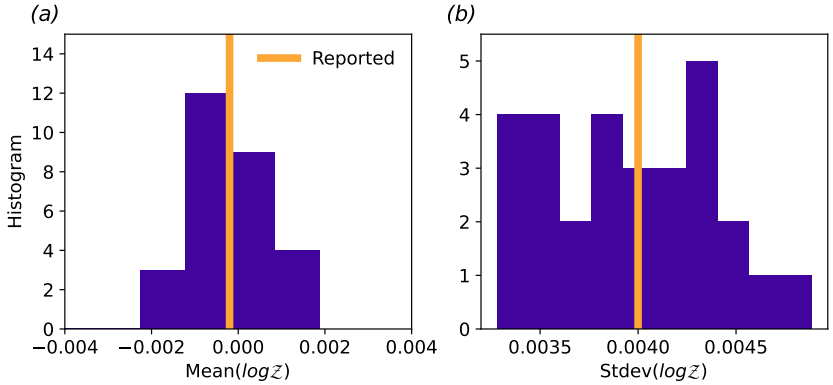

Figure 6. $\log \hat{\mathcal{Z}}$ estimates from 30 independent Liouville flow training for MG (D=2) problem.

Table 7. All $\log \mathcal{Z}$ results of SMC with different tempering scale setting (T).

T	MG ($D=2$)	Funnel ($D=10$)	VAE ($D=30$)	Ionosphere ($D=35$)	Sonar ($D=61$)	LGCP ($D=1600$)
32	-1.29 ± 0.015	-0.17 ± 0.11	-110.91 ± 1.19	-111.84 ± 0.38	-108.41 ± 0.27	408.75 ± 7.27
64	-1.29 ± 0.013	-0.14 ± 0.15	-110.22 ± 0.77	-111.60 ± 0.14	-108.39 ± 0.10	483.00 ± 4.26
128	-1.29 ± 0.010	-0.13 ± 0.061	-110.01 ± 0.74	-111.60 ± 0.059	-108.39 ± 0.04	503.51 ± 1.74
256	-1.29 ± 0.006	-0.12 ± 0.062	-110.20 ± 0.55	-111.62 ± 0.046	-108.39 ± 0.035	506.77 ± 0.68
1024	-1.29 ± 0.004	-0.03 ± 0.17	-110.16 ± 0.41	-111.61 ± 0.026	-108.38 ± 0.018	506.96 ± 0.24

Table 8. All $\log \mathcal{Z}$ results of AFTMC with different tempering scale setting (T).

T	MG ($D=2$)	Funnel ($D=10$)	VAE ($D=30$)	Ionosphere ($D=35$)	Sonar ($D=61$)	LGCP ($D=1600$)
32	2.53 ± 0.05	-0.34 ± 0.13	-110.12 ± 0.48	-86.03 ± 6.95	-99.39 ± 39.49	406.90 ± 7.96
64	2.51 ± 0.05	-0.34 ± 0.079	-110.03 ± 0.39	-118.46 ± 64.28	-97.82 ± 65.72	477.27 ± 4.76
128	2.55 ± 0.05	-0.28 ± 0.10	-110.11 ± 0.41	-113.33 ± 16.61	-98.18 ± 82.91	501.11 ± 2.23
256	2.44 ± 0.05	-0.11 ± 0.68	-110.07 ± 0.36	-121.63 ± 16.37	-104.80 ± 68.31	505.97 ± 1.19

setting $\rho_*(0) = \mathcal{N}(0, I)$. For DDS, we set the σ of the Ornstein–Uhlenbeck process to be 1, which will result in a reference distribution of $\mathcal{N}(0, I)$. For PIS, the uncontrolled reference distribution is determined by the end time of the stochastic process T_{end} and the magnitude of the stochastic process `g_coef`. Using the original settings in the PIS Github repo with `g_coef`=$\sqrt{0.2}$, we set $T_{\text{end}} = 5.0$ to fix the uncontrolled reference distribution as $\mathcal{N}(0, I)$. By doing so, we impose a more challenging and practical set of problems for comparing different methods.

For the Bayesian problems, it is natural to choose the prior distribution as the initial distribution for LFIS. For DDS, to avoid tuning hyper-parameters, we directly use the results provided in the DDS Github repository (Vargas et al., 2023b), which has the same numerical setup for computing the statistics as all other methods. For PIS, we can reproduce the statistics for most of the Bayesian problems by tuning the hyperparameters, except for the LGCP. For completeness, we use the statistics reported in the DDS GitHub repository (Vargas et al., 2023b) for table 9.

D.10. Sequential Monte Carlo

Sequential Monte Carlo (Del Moral et al., 2006) serves both as the reference method as well as the gold standard for those problems that are not analytically tractable. A series of similar studies including AFT (Arbel et al., 2021a), PIS (Zhang & Chen, 2022), and DDS (Vargas et al., 2023a) took the same approach. We found that the SMC implementations in AFT and DDS were unnecessarily complicated. In particular, the leapfrog step size in the Hamiltonian Monte Carlo (HMC) kernel in those implementations depends on the progression of the tempered scales. Although progression-dependent step size was meticulously tuned, we found that a fixed step size with a sufficiently long HMC kernel and sufficiently high Effective

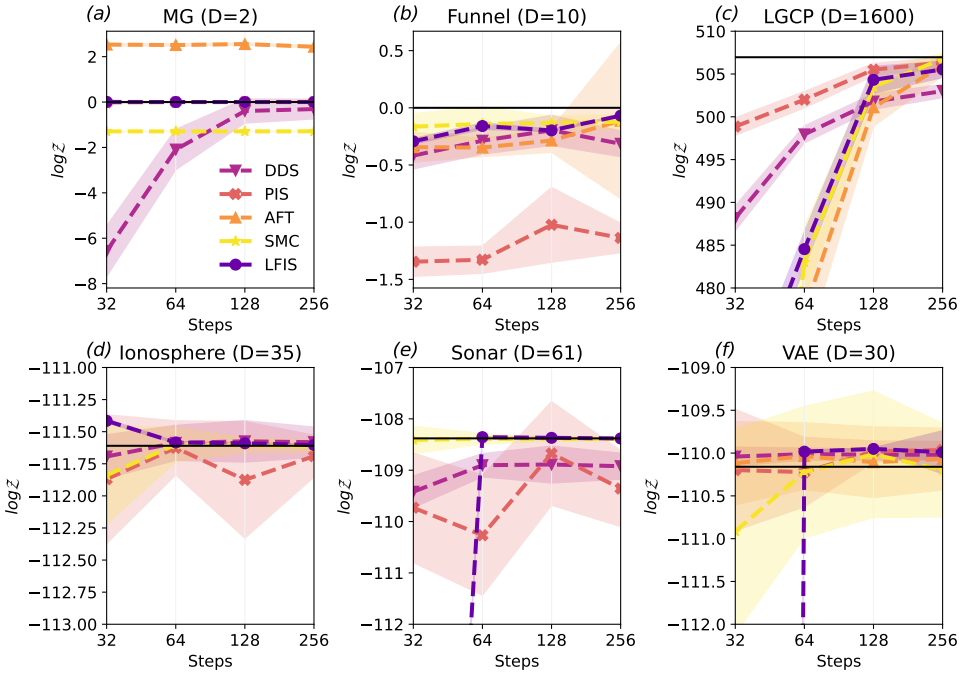

Figure 7. The effects of time steps on $\log \hat{\mathcal{Z}}$ esitmation

Table 9. All $\log \mathcal{Z}$ results of PIS with different tempering scale setting setting (T). The statistics marked by $*$ are results from Vargas et al. (2023a) using the same numerical experimental setup.

T	MG ($D=2$)	Funnel ($D=10$)	VAE ($D=30$)	Ionosphere ($D=35$)	Sonar ($D=61$)	LGCP ($D=1600$)
32	-0.016 ± 0.045	-1.35 ± 0.13	-110.20 ± 0.71	-111.87 ± 0.50	-109.73 ± 1.07	$498.86 \pm 1.02*$
64	-0.003 ± 0.037	-1.33 ± 0.12	-110.22 ± 0.41	-111.63 ± 0.21	-110.27 ± 1.17	$502.00 \pm 0.96*$
128	0.002 ± 0.030	-1.02 ± 0.33	-109.98 ± 0.12	-111.88 ± 0.45	-108.67 ± 1.01	$505.50 \pm 0.72*$
256	0.004 ± 0.018	-1.14 ± 0.13	-109.96 ± 0.10	-111.69 ± 0.16	-109.36 ± 0.74	$506.34 \pm 0.63*$

Table 10. All $\log \mathcal{Z}$ results of DDS with different tempering scale setting setting (T). The statistics marked by $*$ are results from Vargas et al. (2023a) using the same numerical experimental setup.

T	MG ($D=2$)	Funnel ($D=10$)	VAE ($D=30$)	Ionosphere ($D=35$)	Sonar ($D=61$)	LGCP ($D=1600$)
32	-6.57 ± 1.11	-0.42 ± 0.12	$-110.04 \pm 0.11*$	$-111.69 \pm 0.17*$	$-109.41 \pm 0.31*$	$488.17 \pm 1.45*$
64	-2.09 ± 0.89	-0.28 ± 0.12	$-110.01 \pm 0.07*$	$-111.59 \pm 0.14*$	$-108.90 \pm 0.23*$	$497.94 \pm 0.99*$
128	-0.40 ± 0.53	-0.20 ± 0.13	$-110.01 \pm 0.06*$	$-111.58 \pm 0.16*$	$-108.89 \pm 0.36*$	$501.78 \pm 0.90*$
256	-0.31 ± 0.43	-0.31 ± 0.11	$-110.01 \pm 0.06*$	$-111.58 \pm 0.12*$	$-108.92 \pm 0.25*$	$503.01 \pm 0.77*$

Sample Size (ESS) threshold for triggering resampling can achieve similar results. In addition, the code base in Arbel et al. (2021b) also contains an error that the number of HMC kernels per Monte Carlo step (`hmc_steps_per_iter` specified in the configuration files) was not properly performed.

As such, we have streamlined the SMC implementation in this manuscript. The source codes, implemented in PyTorch, will be released if the manuscript is accepted for publication. Our SMC implementation is relatively simple: for each scale, the Monte Carlo step consists of N_H HMC kernels, each of which has L leapfrog steps, and each time step is fixed at δ. Table 12 specifies these hyper-parameters of the SMC for each of the test problems.

We also noticed a nuanced detail about applying SMC on LGCP. AIS and DDS both suggested performing SMC on a whitened representation of the LGCP problem, which was based on diagonalizing the covariance matrix of the problem

Table 11. All $\log \mathcal{Z}$ results of LFIS with different tempering scale setting setting (T).

	T	MG ($D = 2$)	Funnel ($D = 10$)	VAE ($D = 30$)	Ionosphere ($D = 35$)	Sonar ($D = 61$)	LGCP ($D = 1600$)
$\widehat{\log \mathcal{Z}}$	32	-0.064 ± 0.065	-0.43 ± 0.050	-220.70 ± 2.23	-112.28 ± 0.68	-133.36 ± 2.08	470.13 ± 5.57
	64	-0.019 ± 0.060	-0.27 ± 0.066	-110.58 ± 0.39	-112.04 ± 0.38	-108.94 ± 0.30	489.31 ± 3.20
	128	-0.007 ± 0.066	-0.23 ± 0.057	-110.07 ± 0.31	-111.67 ± 0.26	-108.60 ± 0.31	505.26 ± 5.56
	256	0.004 ± 0.059	-0.11 ± 0.075	-109.95 ± 0.25	-111.63 ± 0.33	-108.48 ± 0.30	505.43 ± 3.00
$\log \hat{\mathcal{Z}}$	32	0.006 ± 0.005	-0.29 ± 0.023	-215.64 ± 0.87	-111.42 ± 0.01	-130.45 ± 0.233	463.40 ± 2.61
	64	0.002 ± 0.003	-0.16 ± 0.028	-109.98 ± 0.01	-111.58 ± 0.006	-108.36 ± 0.011	484.54 ± 2.01
	128	-0.0001 ± 0.004	-0.20 ± 0.011	-109.95 ± 0.01	-111.59 ± 0.004	-108.37 ± 0.008	504.32 ± 1.56
	256	-0.0002 ± 0.004	-0.07 ± 0.003	-109.99 ± 0.08	-111.60 ± 0.006	-108.38 ± 0.009	505.53 ± 0.95

Table 12. Specification of our SMC implementations.

	MG	Funnel	VAE	Ionosphere	Sonar	LGCP
δ	0.02	0.02	0.2	0.02	0.02	0.2
L	20	20	10	20	20	20
N_H	10	10	2	10	10	2
ESS threshold	0.98	0.98	0.98	0.98	0.98	0.98

by Cholesky decomposition. However, we found that for the tempering scale $T \gtrsim 256$, a whitened representation is not necessary. We provide a comparison of applying SMC on the whitened and un-whitened representations in Table 13. As such, we only reported the SMC results on the unwhitened LGCP problem in the rest parts of our manuscript.

Table 13. SMC estimation of $\log \mathcal{Z}$ on whitened and unwhitened representation of the Log-Gaussian Cox Problem.

T	Whitened	Unwhitened
32	506.90 ± 0.12	408.75 ± 7.27
64	506.89 ± 0.09	483.00 ± 4.26
128	506.87 ± 0.068	503.51 ± 1.74
256	506.89 ± 0.049	506.77 ± 0.68
1024	506.90 ± 0.018	506.96 ± 0.24

E. Ablation study of Annealed Flow Transport Monte Carlo Sampler

We tested the AFTMC sampler (with 256 scales) without the MC kernel on the Gaussian Mixture, Funnel, Variational Auto-Encoder, and Log-Gaussian Cox Process problems. We juxtapose the results of (1) SMC with $T = 1024$ as the gold standard, (2) SMC with $T = 256$ as a reference, (3) AFT with a Monte Carlo kernel and $T = 256$, and (4) AFT without the Monte Carlo kernel and $T = 256$ in Table 14. The results indicate that the MC kernel, instead of the flow-transport kernel, is the major functioning component in estimating the marginal likelihood $\log \mathcal{Z}$. The AFTMC algorithm with only the MC kernel is functionally identical to SMC. The numerical experiments on these datasets seemed to suggest that the flow transport implemented as a normalizing flow, parametrized by variational inference, does not seem to function as expected.

Table 14. Ablation study on the effect of Monte Carlo kernel in Annealed Flow Transport Monte Carlo sampler (Arbel et al., 2021a) estimating $\log \mathcal{Z}$.

Model	MG ($D = 2$)	Funnel ($D = 10$)	VAE ($D = 30$)	LGCP ($D = 1600$)
SMC (1024)	-1.29 ± 0.0046	-0.034 ± 0.17	-110.16 ± 0.41	506.96 ± 0.24
SMC (256)	-1.29 ± 0.0062	-0.12 ± 0.062	-110.20 ± 0.54	506.77 ± 0.68
AFT with MC (256)	2.44 ± 0.05	-0.11 ± 0.68	-110.07 ± 0.36	505.97 ± 1.19
AFT without MC (256)	2.94 ± 0.39	-0.79 ± 0.40	-288.18 ± 58.28	-826.73 ± 8.96

Table 15. Adjustable meta-parameters and objects in a range of methods. Markers ✓ indicate the adjustable objects that the user must specify. We remark that within the MC kernel there could be multiple adjustable parameters, e.g., see Sec. D.10 and Table 12.

Adjustable parameters/objects	SMC	AFTMC	CR-AFTMC	PIS	DDS	LFIS
Number of time steps/scales T	✓	✓	✓	✓	✓	✓
Terminal time				✓		
Neural network (per time step/scale)		✓	✓	✓	✓	✓
Monte Carlo kernel (per time step/scale)	✓	✓	✓			
Reference process $\tilde{\rho}_*(x, t)$				✓	✓	
Annealed path of distributions $\tilde{\rho}_*(x, t)$	✓	✓	✓			✓
Resampling threshold	✓	✓	✓			
Initial distribution	✓	✓	✓			✓