Learning Deformation Trajectories of Boltzmann Densities

Anonymous authors
Paper under double-blind review

Abstract

We introduce a training objective for continuous normalizing flows that can be used in the absence of samples but in the presence of an energy function. Our method relies on either a prescribed or a learnt interpolation f_t of energy functions between the target energy f_1 and the energy function of a generalized Gaussian $f_0(x) = ||x/\sigma||_p^p$. The interpolation of energy functions induces an interpolation of Boltzmann densities $p_t \propto e^{-f_t}$ and we aim to find a time-dependent vector field V_t that transports samples along the family p_t of densities. The condition of transporting samples along the family p_t can be translated to a PDE between V_t and f_t and we optimize V_t and f_t to satisfy this PDE. We experimentally compare the proposed training objective to the reverse KL-divergence on Gaussian mixtures and on the Boltzmann density of a quantum mechanical particle in a double-well potential.

1 Introduction

We consider the task of learning and sampling from a probability density p proportional to the unnormalized density e^{-f} where $f : \mathbb{R}^n \to \mathbb{R}$ is a given energy function. In particular, we don't have access to samples from p, all we have is the ability to evaluate f and its derivatives for any $x \in \mathbb{R}^n$. A popular technique (Boyda et al., 2021; Albergo et al., 2021a;b; 2022; Abbott et al., 2022; de Haan et al., 2021; Gerdes et al., 2022; Noé et al., 2018; Köhler et al., 2020; Nicoli et al., 2020; 2021) for attacking this problem is to use a normalizing flow to parametrize a density q_θ and optimize the parameters θ to minimize the reverse KL-divergence

$$KL[q_\theta, p] = \mathbb{E}_{x\sim q_\theta}(\log q_\theta(x) - \log p(x)) = \mathbb{E}_{x\sim q_\theta}(\log q_\theta(x) + f(x)) + Z. \tag{1}$$

The use of normalizing flows for this problem is particularly attractive because a normalizing flow realizes a probability distribution q_θ that can be used as a proposal for importance sampling to account for the inaccuracies of q_θ. Unfortunately, the reverse KL-divergence is mode-seeking, making the training prone to mode-collapse (Fig. 1). To tackle this problem, several works have proposed alternative training objectives for normalizng flows. Vaitl et al. (2022) introduce estimators of the forward KL divergence, while Midgley et al. (2022) replace the reverse KL-divergence with the $\alpha = 2$ divergence and combine it with annealed importance sampling (Neal, 1998). We propose yet another alternative based on infinitesimal deformations of Boltzmann densities (Pfau & Rezende, 2020; Máté & Fleuret, 2022).

The contributions of this work can be summarized as follows

- In §3 we describe our method which relies on either a prescribed or a learnt interpolation f_t of energy functions between the target energy f_1 and the energy function of a generalized Gaussian $f_0(x) = ||x/\sigma||_p^p$. Given f_t we optimize a vector field V_t to transport samples along the family $p_t(x) \propto e^{-f_t(x)}$ of Boltzmann densities. After translating this condition to a PDE between V_t and f_t we propose to minimize the amount by which this PDE fails to hold.

  - First we find that in certain cases the linear interpolation $f_t = (1-t)f_0 + tf_1$ already leads to improved performance over optimizing the reverse KL-divergence. We also show that in general this interpolation is insufficient.
  - Motivated by the failure mode of the linear interpolation, we parametrize the interpolation with another neural network $f_t = (1-t)f_0 + tf_1 + t(1-t)f^\theta(t)$ and optimize f_t^θ together with the vector field V_t^θ.
- In §4 we run experiments on Gaussian mixtures and on the Boltzmann density of a quantum particle in a double-well potential, and report improvements in KL-divergence, effective sample size, and mode coverage.

Motivation

Consider the following multimodal density

$$\frac{1}{4}\Big(\mathcal{N}\left(\begin{bmatrix}-8 & -8\end{bmatrix},1\right)+\mathcal{N}\left(\begin{bmatrix}-8 & 8\end{bmatrix},1\right)+\mathcal{N}\left(\begin{bmatrix}8 & -8\end{bmatrix},1\right)+\mathcal{N}\left(\begin{bmatrix}8 & 8\end{bmatrix},1\right)\Big), \tag{2}$$

where $\mathcal{N}(\mu,\sigma)$ denotes a normal density centered at μ with covariance matrix $\mathrm{diag}(\sigma^2)$. Fig. 1 shows the mode-collapse of a normalizing flow trained with the reverse KL-divergence on this target. The reason why mode collapse can happen in the first place is the training objective itself. The mode seeking behavior of the reverse KL-divergence can be explained as follows. As the sampling is done according to q_θ, the difference in log-likelihoods is weighted by the likelihood q_θ. This implies that if q_θ completely ignores modes of p the log-likelihoods of p and q_θ are not compared over regions that are not covered by q_θ. In this paper we introduce a training objective for continuous normalizing flows that can be used to replace the reverse KL-divergence and investigate to what extent it solves the issue of mode collapse on multimodal targets.

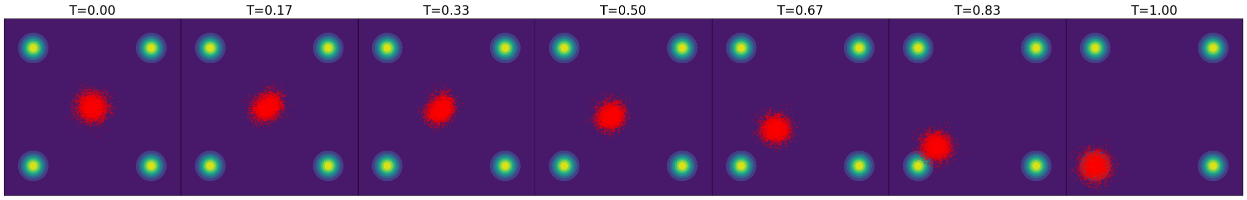

Figure 1: Mode-seeking nature of the reverse KL-divergence. The figures from left to right show how the latent gaussian is transformed by the continuous normalizing flow trained with the reverse KL objective. The green blobs denote the (unnormalized) target density, given by Eq. 2.

2 Background

Change of variables

Let $p_0(z)$ be a probability density on $\mathbb{R}^n$ and $\Psi : \mathbb{R}^n \to \mathbb{R}^n$ a diffeomorphism. Pushing the density p_0 forward along Ψ induces a new probability density p implicitly defined by

$$\log p_0(z) = \log p(\Psi z) + \log|\det J_\Psi(z)|. \tag{3}$$

The term $\log|\det J_\Psi(z)|$ measures how much the function Ψ expands volume locally at z.

Continuous change of variables. Let now V_t be a time-dependent vector field and Ψ_τ denote the diffeomorphism of following the trajectories of V_t from 0 to τ. This family of diffeomorphisms generates a one-parameter family of densities p_τ. The amount of volume expansion a particle experiences along this trajectory of between 0 and τ is $\int_0^\tau \nabla \cdot V_t(\Psi_t z)dt$. The log-likelihoods are then related by

$$\log p_0(z) = \log p_\tau(\Psi_\tau z) + \int_0^\tau \nabla \cdot V_t(\Psi_t z)dt. \tag{4}$$

Normalizing flows

Normalizing flows (Tabak & Turner, 2013; Dinh et al., 2016) (continuous normalizing flows (Chen et al., 2018)) parametrize a subset of the space of all densities on $\mathbb{R}^n$. They do this by first fixing a base density p_0 and using a neural network that parametrizes the transformation Ψ (the vector field V_t). The change of variables formula in Eq. 3 (Eq. 4) is then applied to compute the density induced by Ψ (V_t). Generalizations of normalizing flows have also been gaining popularity in the recent years (Nielsen et al., 2020; Huang et al., 2020; Máté et al., 2022).

Boltzmann densities

Let $f : \mathbb{R}^n \to \mathbb{R}$ be an energy function[1] with a finite normalizing constant $Z = \int e^{-f(x)} d^n x$. The function f then induces a Boltzmann density over the configurations $x \in \mathbb{R}^n$,

$$p(x) = \frac{1}{Z} e^{-f(x)}. \tag{5}$$

Conversely, given a probability density function $p : \mathbb{R}^n \to \mathbb{R}_{+,0}$ the corresponding energy function can be recovered up to a constant $f = -\log p - \log Z$. In somewhat more abstract terms, the space of probability densities is obtained by factoring the space of energy functions by $\mathbb{R}_+$. Throughout the rest of the paper we will keep making use of this "almost equivalence" between densities and energy functions, and will frequently translate between the density space and log-density space.

The deformation equation

A probability density $p_0 \propto e^{-f_0}$ and a one-parameter family of vector field V_t generates a one-parameter family of probability densities $p_t \propto e^{-f_t}$ simply by making the samples from p_0 follow the flow[2] determined by V_t. Conversely, given a one-parameter family of probability densities p_t, we are interested in finding the *transport field* or *deformation field* V_t. Once V_t is found, it can be used to transform samples between different members of the family p_t. Either way, the families f_t and V_t are related by

$$\partial_t f_t + \langle \nabla f_t, V_t \rangle - \nabla \cdot V_t + C_t = 0, \tag{6}$$

where ∇f_t denotes the gradient of f_t, $\nabla \cdot V_t$ the divergence of V_t, $\langle \, , \, \rangle$ is the Euclidean scalar product between vectors, and C_t is a spatially constant function, i.e. only depends on time. Moreover, if we find a triplet (f_t, V_t, C_t) that solves Eq. 6, then C_t necessarily equals $\partial_t \log Z_t$ (up to an additive constant). We refer the reader to the works of Pfau & Rezende (2020, Eq. 6) and Máté & Fleuret (2022, Eq. 16) for details and Fig 2 for an example.

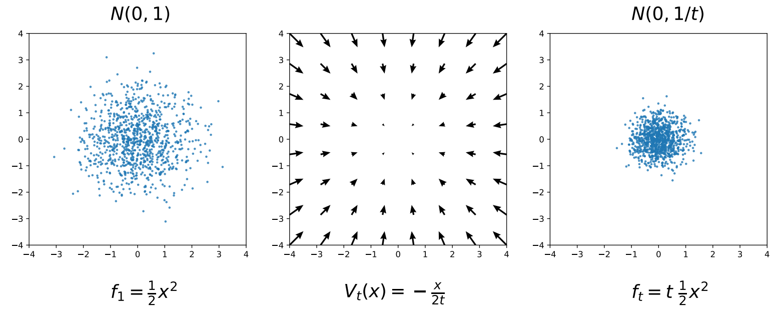

Figure 2: Deformation of a two-dimensional standard Gaussian into a family of isotropic Gaussians with different variances. Samples from a two-dimensional standard Gaussian (left). The deformation field (center). The samples in the left plot flowing along the deformation field from 1 to t follow a Gaussian centered at the origin with covariance $1/t$ (right).

---

[1] Depending on the field, other authors might call this *action* or *potential*.

[2] The word *flow* is not used here in the usual sense in a machine learning context, but rather in the sense of differential geometry.

3 Approximating the transport field

From here on, we will use the vector field V_t and the term “continuous normalizing flow” interchangeably. Our goal is to sample from a target Boltzmann density $p_{target} \propto e^{-f_{target}}$ by

1. defining a family of energy functions f_t, $0 \leq t \leq 1$ interpolating between the target energy $f_1 = f_{target}$ and the energy function of a generalized Gaussian $f_0(x) = ||x/\sigma||_p^p$,
2. finding a transport field V_t such that (f_t, V_t) “solves” the deformation equation (Eq. 6).

If we succeed at both of these constructions, then we can obtain samples from p_{target} by sampling from $p_0 \propto e^{-||x/\sigma||_p^p}$ and let the samples follow the flow of V_t from $t = 0$ to $t = 1$. Note that we turned the problem of learning a single density e^{-f_1} into a continuous collection of problems, learning all the densities e^{-f_t} for $0 \leq t \leq 1$. It might seem like that we just made the task more difficult, but the idea is that for any $0 \leq \tau \leq 1$, the density e^{-f_τ} is easier to fit once $e^{-f_{\tau-\varepsilon}}$ is already fitted for some small ε.

The pointwise deformation error

Regarding the second item of the above list, an analytical expression for V_t is not easy to find if we are given a family of energy functions f_t. This would amount to solving Eq. 6, which is difficult in general. Therefore we will parametrize V_t with a neural network and train it to minimize the amount by which the pair (f_t, V_t) fails to satisfy the deformation equation. We begin by recalling the deformation equation,

$$\partial_t f_t + \langle \nabla f_t, V_t \rangle - \nabla \cdot V_t + C_t = 0, \tag{7}$$

where C_t is a spatially constant function. In what follows, V_t^θ and C_t^θ are parametrized by neural networks and are trained to minimize some monotonically increasing function L[3] of the pointwise *deformation error*

$$\mathcal{E}_{\theta,x,t} = |\partial_t f_t(x) + \langle \nabla f_t(x), V_t^\theta(x) \rangle - \nabla \cdot V_t^\theta(x) + C_t^\theta|. \tag{8}$$

The expression $L(\mathcal{E}_{\theta,x,t})$ measures the incompatibility of f_t and V_t at a single (t, x) pair of coordinates, we will need to optimize some sort of integral of this pointwise error over both t and x.

The deformation loss

Suppose that we have an interpolation of energy functions f_t. We propose to train V_t^θ and C_t^θ to minimize the deformation error (Eq. 8) along the trajectories of V_t^θ. Formally, let q_θ be a parametric density parametrized by a continuous normalizing flow V_t^θ. We update the parameters to minimalize the integral of $L(\mathcal{E})$ along the trajectories of the flow,

$$\mathcal{L}(\theta) = \mathbb{E}_{z\sim\mathcal{B}} \left[w(z) \int_0^1 L\left(\mathcal{E}_{\theta,\gamma_t^\theta(z),t}\right) dt \right], \tag{9}$$

where $\mathcal{B}$ denotes the base distribution and $\gamma_\tau^\theta(z)$ is given by transporting z along the V_t^θ between 0 and τ. The weights $w(z)$ in Eq. 9 are estimates of the importance weights $\frac{e^{-f(\gamma_1(z))}/Z}{q_\theta(\gamma_1(z))}$. They are computed from the unnormalized importance weights $\tilde{w}_i = \frac{e^{-f(\gamma_1(z_i))}}{q_\theta(\gamma_1(z_i))}$ as $w(z_i) = \tilde{w}_i / \left(\frac{1}{N}\sum_{j=1}^N \tilde{w}_j\right)$ over a batch of samples. We evaluate the integral in Eq. 9 by discretizing time and using numerical ODE solvers. The standard deviation of the base is an important hyperparameter, its role can be explained as follows. Ideally, we would like to minimize the deformation error everywhere in space, but we can only evaluate it along the trajectories of V_t^θ. To provide better coverage, we can increase the standard deviation of the base density during training. In principle, it is not a requirement to train along trajectories starting from the same base as the one we're trying to match to the target. We didn't explore this possibility, in all our experiments $\mathcal{B}$ is both the distribution along which we optimize the deformation loss and also the base of the flow (i.e. the density that approximates p once pulled through the flow).

---

[3] In our experiments we tried $L : \mathbb{R}^+ \to \mathbb{R}^+ \in \{|\mathcal{E}| \mapsto |\mathcal{E}|, |\mathcal{E}| \mapsto |\mathcal{E}|^2, |\mathcal{E}| \mapsto |\mathcal{E}| + |\mathcal{E}|^2, |\mathcal{E}| \mapsto \log(1 + |\mathcal{E}|)\}$.

First approach: Linear interpolation

There are many ways of interpolating between two functions. Arguably the simplest way to do so is to set

$$f_t(x) = (1-t)f_0(x) + t f_1(x), \qquad 0 \le t \le 1. \tag{10}$$

A discretization of the induced interpolation in density space is often used as a sequence of proposals for annealed importance sampling (Neal, 1998). We will call the family given by Eq. 10 the *linear interpolation*. Fig. 3 shows the evolution of the samples along a continuous normalizing flow trained with the deformation loss using the linear interpolation. Comparing with Fig. 1, we observe that the same network trained with the deformation loss instead of the reverse KL-divergence can capture all 4 modes of the target density.

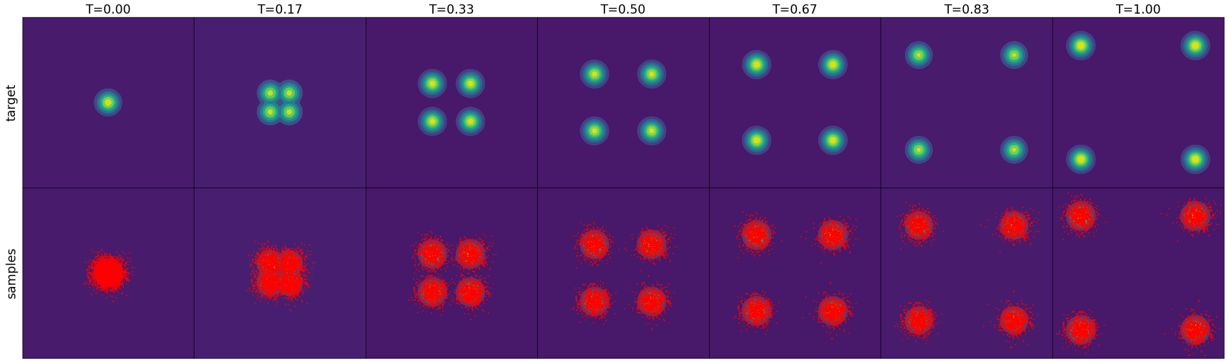

Figure 3: The same continuous normalizing flow as in Fig. 1 trained with the deformation loss using the linear interpolation. The top row shows how the target density evolves under the predefined linear interpolation between $f_0 = x^2/2$ and $f_1 = -\log(\text{Eq. 2})$, while the bottom row shows how the samples from q_θ evolve along V_t as t is varied.

The issue with the linear interpolation. Let us now consider the density

$$\tfrac{1}{3}\mathcal{N}\left(\begin{bmatrix}4 & 4\end{bmatrix}, 1\right) + \tfrac{2}{3}\mathcal{N}\left(\begin{bmatrix}-8 & -8\end{bmatrix}, 1\right). \tag{11}$$

This target does not enjoy the symmetry properties of the previous mixture, one of the modes is closer to the base and has a lower relative weight than the other one. The top row of Fig. 4 shows the linear interpolation to this target and the second row shows that this interpolation is insufficient to capture the mode which is further away. In a nutshell, the reason for this is that the relative weights of the modes are not preserved as t is varied.

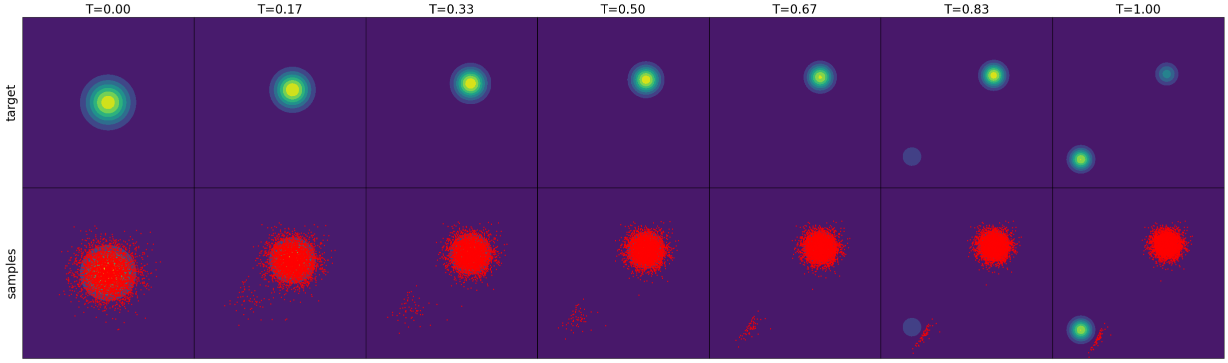

Figure 4: The issue with the linear interpolation. The top row shows how the target density evolves under the predefined linear interpolation between $f_0 = x^2/8$ and $f_1 = -\log(\text{Eq. 11})$, while the bottom row shows how the samples from q_θ evolve along V_t as t is varied.

Interlude: good and bad interpolations

The reason why the linear interpolation can lead to problems is illustrated in Figures 5 and 6. Figure 5 shows a particular example where the linear interpolation in log-density space induces a “good” interpolation in density space. Figure 6 shows that for certain target functions the linear interpolation in log-density space induces an interpolation in density space that is not “local” in the sense that probability mass gets moved between the modes. Such densities are difficult to learn with the linear interpolation, even if in principle there exists a vector field V_t that moves probability mass the correct way along any interpolation.

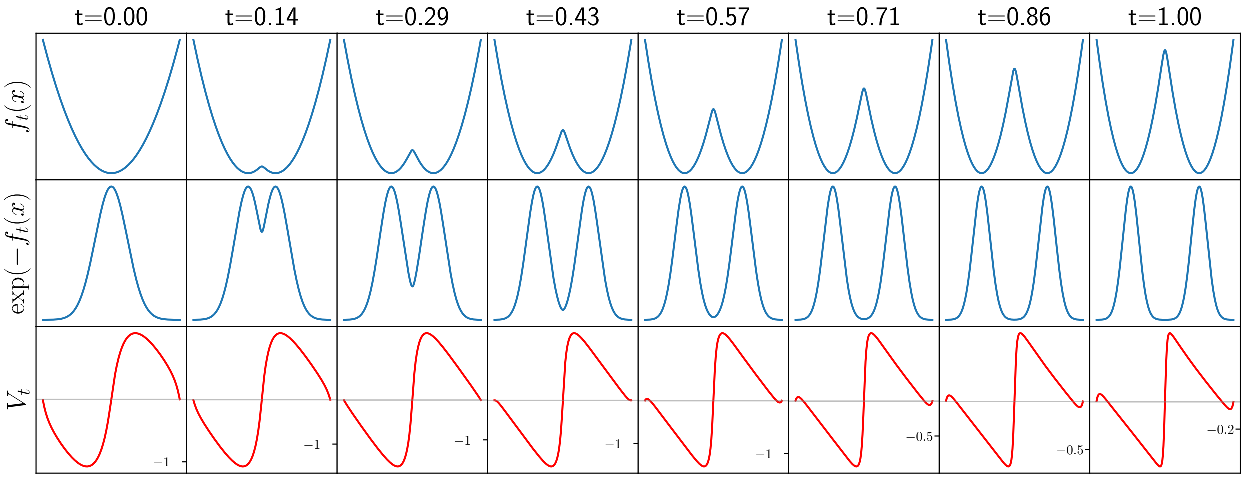

Figure 5: Linear interpolation between $f_0(x) = x^2/4$ and $f_1 = -\log(\frac{1}{2}e^{-(x-3)^2} + \frac{1}{2}e^{-(x+3)^2})$. Linear interpolation in the log-density space (top row), and the induced interpolation in density space (middle row). The transport field V_t which can be represented as a scalar-valued function for one-dimensional densities (bottom row). The thin gray line in the bottom row denotes $V_t = 0$.

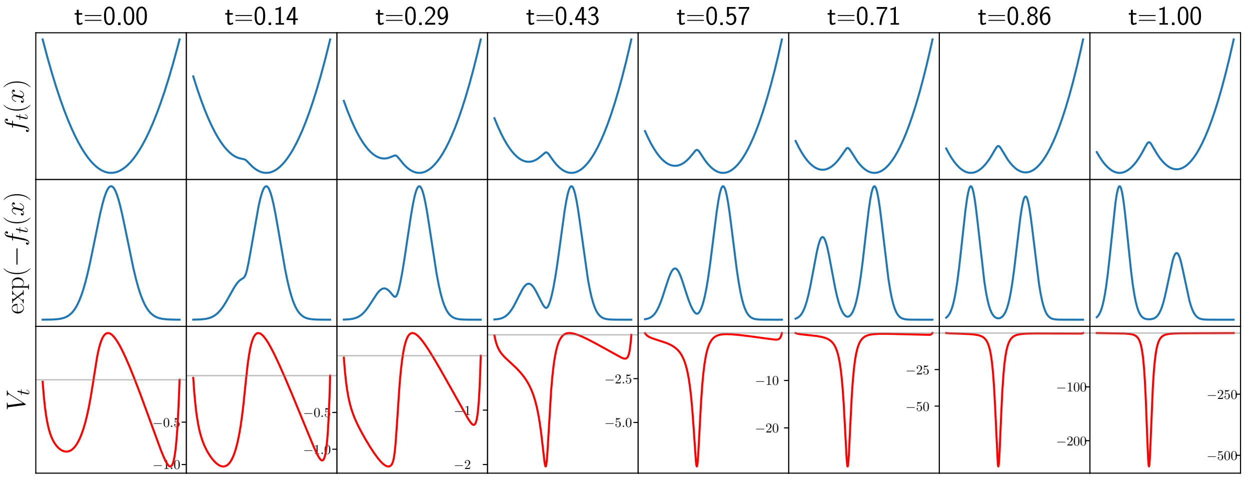

Figure 6: Linear interpolation between $f_0(x) = x^2/4$ and $f_1 = -\log(\frac{1}{3}e^{-(x-1)^2} + \frac{2}{3}e^{-(x+4)^2})$. Linear interpolation in the log-density space (top row), and the induced interpolation in density space (middle row). The transport field V_t which can be represented as a scalar-valued function for one-dimensional densities (bottom row). The thin gray line in the bottom row denotes $V_t = 0$. Note how the relative weights of modes is not preserved as t is varied, and the exploding norm of V_t.

Sanity check: interpolate along the diffusion process

Diffusion Models. Diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020) also define a family of probability densities p_t on $\mathbb{R}^n$ interpolating between the data density $p_1(x)$ and the latent $n-$dimensional gaussian $p_0 = \mathcal{N}(0, \sigma)$. Samples from p_t are of the form $\sqrt{t}x + \sqrt{1-t}z$ where $x \sim p_1, z \sim p_0$. Geometrically, the probability density p_t is obtained by first stretching p_1 by a factor of $\sqrt{t}$ and then convolving with the Gaussian kernel $\mathcal{G}^{\sigma\sqrt{1-t}4}$.

$$p_1(x) \xrightarrow{\lambda_t} t^{-n/2} p_1(t^{-1/2}x) \xrightarrow{G_t} \underbrace{t^{-n/2} p_1(t^{-1/2}x) * \mathcal{G}^{\sigma\sqrt{1-t}}}_{p_t(x)}, \tag{12}$$

where we used λ_t to denote the stretching by a factor of $\sqrt{t}$ and G_t to denote the convolution with the Gaussian kernel $\mathcal{G}^{\sigma\sqrt{1-t}}$. If $t = 1$, both λ_t (stretching by a factor of 1) and G_t (convolving with a Dirac-delta) are just the identity. If $t = 0$, then λ_t collapses p_1 to a Dirac-delta at the orgin and G_t convolves it with $\mathcal{N}(0, \sigma)$ implying that $p_0 = \mathcal{N}(0, \sigma)$. The nice thing about the diffusion process is that it provides an interpolation between densities where the situation of Fig. 6 is avoided. Loosely speaking, this interpolation is "local" in a sense that that the linear interpolation was not.

The family of Gaussian mixtures is closed under the diffusion process given by Eq. 12, we can even explicitly compute the time-evolution of a Gaussian mixture and use it to replace the linear interpolation. Fig. 7 shows samples from a normalizing flow that was trained with the deformation loss using the hand-computed interpolation of its diffusion process.

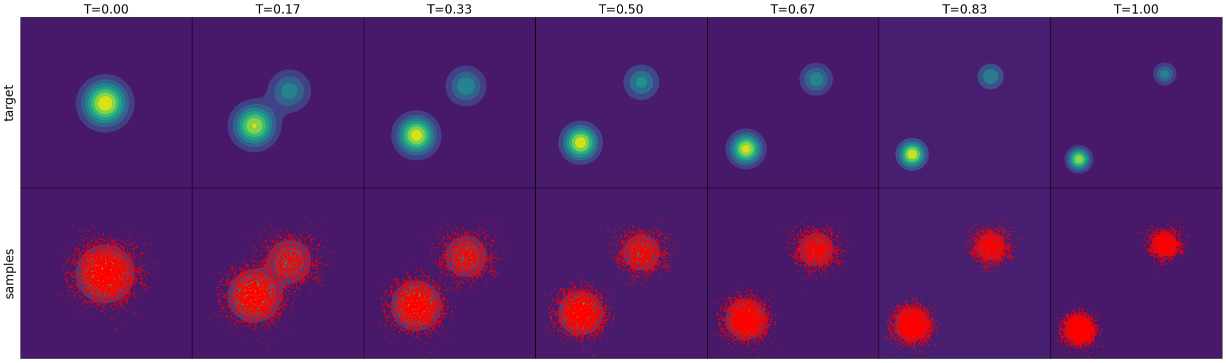

Figure 7: The same continuous normalizing flow as in Fig. 4 with the deformation loss using the hand-computed diffusion interpolation. The green blobs denote the (unnormalized) target density, a mixture of two Gaussians, given by Eq. 11. The top row shows how the target density evolves under the predefined diffusion-like interpolation, while the bottom row shows how the samples from q_θ evolve along the flow as t is varied.

Unfortunately, since all we are given is the ability to evaluate the target energy pointwise, there is no analytical formula for calculating the diffusion process of an arbitrary target density. Moreover, any numerical approach involves integration, that gets increasingly expensive as t gets smaller and the dimensionality of the problem gets larger. Nonetheless, this result demonstrates that minimizing the deformation error is a feasible approach, one just needs to be more careful about the interpolation between the latent and target energy functions.

Parametrizing the interpolation

To circumvent the issues presented in the previous section, we need to use a different interpolation between the latent and target energy functions. The authors could not find a predefined interpolation that is 1)

---

[4] The Gaussian kernel $\mathcal{G}^{\sigma\sqrt{1-t}}$ is a Gaussian density with mean 0 and covariance matrix $\mathrm{diag}(\sigma^2(1-t))$.

“local” in the still not formal, but intuitive sense and 2) easy to compute for arbitrary target energies. Instead, we propose to use a neural network to parametrize the interpolation f_t as

$$f_t(x) = (1-t)f_0(x) + tf_1(x) + t(1-t)f_t^\theta(x), \qquad 0 \leq t \leq 1, \tag{13}$$

where f_t^θ is parametrized by a neural network. This parametrization ensures that the boundary conditions at $t \in \{0,1\}$ are satisfied, and allows for flexibility for $0 < t < 1$. The parameters of the interpolation are trained together with the those of the flow (and those of C_t) with the objective of minimizing the deformation loss. Fig. 8 shows that this flexibility allows a flow trained with the deformation loss to capture both modes of the distribution given by Eq. 11.

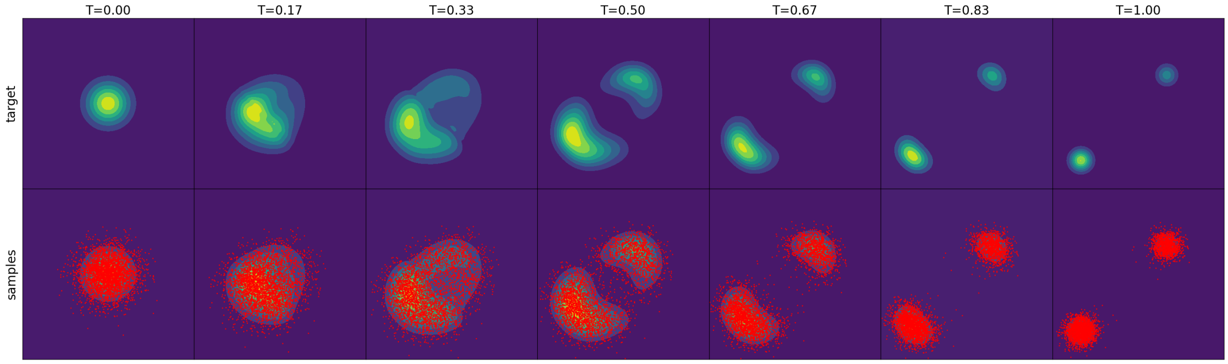

Figure 8: The same continuous normalizing flow as in Fig. 4 trained with the deformation loss using the trainable interpolation. The green blobs denote the (unnormalized) target density, a mixture of 2 Gaussians, given by Eq. 11. The top row shows how the target density evolves under the learned interpolation, while the bottom row shows how the samples from q_θ evolve along V_t as t is varied.

4 Experiments

To compare the deformation loss to the reverse KL-divergence we train the same normalizing flow architecture by minimizing the 1) reverse KL objective 2) the deformation loss. We test on Gaussian mixtures and on the Boltzmann distribution of a quantum mechanical particle in a double-well potential.

Performance metrics

To quantify the results of the experiments, for each model we report a subset of the following metrics. For all runs we report the reverse KL-divergence (minus $\log Z$),

$$\mathbb{E}_{x\sim q_\theta}(\log q_\theta(x) - \log p(x)) \tag{14}$$

and the effective sample size,

$$ESS_r = \frac{\left(\frac{1}{N}\sum_i p(x_i)/q_\theta(x_i)\right)^2}{\frac{1}{N}\sum_i (p(x_i)/q_\theta(x_i))^2}, \qquad x_i \sim q_\theta, \tag{15}$$

where N is the number of samples. These metrics capture how good a fit q_θ is for p, but only in those regions where samples are available. They are therefore insensitive to mode collapse. To compensate for this, we compute the Hausdorff distance between the means of the modes $M = \{m_1, ..., m_k\}$ and a batch of samples $X = \{x_1, ...x_N\}$ from the model,

$$H(M, X) = \max_{m\in M} \min_{x\in X} \sqrt{||m-x||^2}. \tag{16}$$

In the case of Gaussian mixtures, the means M are the means of mixture components whereas in the case of the quantum mechanical particle, the means are $(\phi_1, \phi_2, ..., \phi_N) = (a, a, ..., a)$ and $(\phi_1, \phi_2, ..., \phi_N) = (b, b, ..., b)$

where a and b are the two local minima of $V(/phi) = -m\phi^2 + \lambda\phi^4$ (see §4.2 for details). The Hausdorff distance is a good metric for measuring mode coverage but is insensitive to the shape of the distributions. Finally, the forward KL-divergence (plus $\log Z$),

$$\mathbb{E}_{x\sim p}(\log p(x) - \log q_\theta(x)) \tag{17}$$

and effective sample size,

$$ESS_f = \frac{\left(\frac{1}{N}\sum_i q_\theta(x_i)/p(x_i)\right)^2}{\frac{1}{N}\sum_i (q_\theta(x_i)/p^\theta(x_i))^2}, \qquad x_i \sim p. \tag{18}$$

provide the most accurate (both mode coverage and shape matching) description of the goodness of the fit. As they require samples from p, we only report them for the experiments on Gaussian mixtures.

4.1 Gaussian mixtures

In this section we train on two energy functions, those given by Equations 2 and 11. The metrics are reported in Table 1 and correspond well to what we observe in Figures 1, 3, 4 and 8.

Energy function given by Eq. 2					
	$H(M,X)\downarrow$	Rev. KL $\downarrow$	Rev. ESS $\uparrow$	Fw.KL $\downarrow$	Fw. ESS $\uparrow$
$KL(q_\theta, p)$	$19.26 \pm .333$	$1.387 \pm .001$	$0.999 \pm .000$	110.6 ± 97.6	$0.254 \pm .006$
Def. Loss with Linear Int.	$\mathbf{0.060} \pm .023$	$\mathbf{0.003} \pm .002$	$0.998 \pm .001$	$\mathbf{0.001} \pm .001$	$\mathbf{0.998} \pm .002$
Def. Loss with Trainable Int.	$\mathbf{0.053} \pm .010$	$\mathbf{0.000} \pm .001$	$1.000 \pm .000$	$\mathbf{0.000} \pm .000$	$\mathbf{0.999} \pm .000$
PathPQ Vaitl et al. (2022)	$10.83 \pm .553$	$0.757 \pm .018$	$0.967 \pm .003$	38.25 ± 4.99	$0.460 \pm .010$
Energy function given by Eq. 11					
	$H(M,X)\downarrow$	Rev. KL $\downarrow$	Rev. ESS $\uparrow$	Fw.KL $\downarrow$	Fw. ESS $\uparrow$
$KL(q_\theta, p)$	$13.25 \pm .363$	$1.099 \pm .000$	$1.000 \pm .000$	74.38 ± 30.3	$0.332 \pm .007$
Def. Loss with Linear Int.	$2.329 \pm .124$	$1.081 \pm .004$	$0.928 \pm .080$	37.68 ± 6.99	$0.336 \pm .008$
Def. Loss with Trainable Int.	$\mathbf{0.045} \pm .022$	$\mathbf{0.001} \pm .001$	$0.999 \pm .000$	$\mathbf{0.000} \pm .000$	$\mathbf{1.000} \pm .000$
PathPQ Vaitl et al. (2022)	$0.041 \pm .012$	$0.043 \pm .022$	$0.978 \pm .002$	$0.015 \pm .004$	$0.859 \pm .174$

Table 1: Results of training the same flow with 3 different objectives: the reverse KL-divergence, the deformation loss with linear interpolation and the deformation loss with the trainable interpolation. Note that the target densities are normalized, i.e. $\log Z = 0$. Mean and standard deviation values over 5 seeds are reported.

4.2 Quantum mechanical particle in a double-well potential

In this section we consider the trajectory of a quantum mechanical particle moving in a double-well potential. The action associated to a discretised trajectory $(\phi_1, \ldots, \phi_N)$ is

$$S(\phi_1, \ldots, \phi_N) = \sum_{i=1}^{N} \Big((\phi_i - \phi_{i+1})^2 + V(\phi_i) \Big), \tag{19}$$

where the subscript $i+1$ is to be understood modulo N. We assume that the potential V is of the form

$$V(\phi_i) = -m\phi_i^2 + \lambda\phi_i^4, \tag{20}$$

where m and λ are numerical parameters. In all our experiments $\lambda = 1/16, N \in \{4, 8, 16, 32, 64\}$ and $m \in \{0.25, 0.50, 0.75, 1.00\}$. The Boltzmann density then reads

$$e^{-S(\phi_1, \ldots, \phi_N)} = \prod_{i=1}^{N} e^{-(\phi_i - \phi_{i+1})^2} \prod_{i=1}^{N} e^{-V(\phi_i)}. \tag{21}$$

Heuristics. In all cases, the one-dimensional Boltzmann density $e^{-V(\phi)}$ has two modes and their separation is controlled by m, (Fig. 9). Intuitively, the second factor in Eq. 21 encourages the particle to follow the unnormalized density $e^{-V(\phi)}$ at every time step, while the first one penalizes if the values of ϕ at consecutive time steps differ too much (i.e. belong to different modes of $e^{-V(\phi)}$). With the above interpretation we can argue that ϕ_i and ϕ_{i+1} are likely to be close, and every ϕ_i is likely to belong to one of the modes of $e^{-V(\phi_i)}$. Then the density $e^{-S(\phi_1,\ldots,\phi_N)}$ has two modes, centered at $(a, a, ..., a)$ and $(b, b, ..., b)$, where a and b are the local minima of V.

Sensitivity to the mass parameter m. Now we fix $N = 16$ and vary the mass of particle $m \in \{0.25, 0.50, 0.75, 1.00\}$. We compare the deformation loss with the trainable interpolation to the reverse KL objective. The quantitative results are summarized in Table 2. In Figure 9 we compare $e^{-V(\phi)}$ to the histogram of flattened samples from the trained models. This makes sense since the action encourages the particle to follow the one-dimensional Boltzmann density of the potential $V(\phi)$ at every time-step.

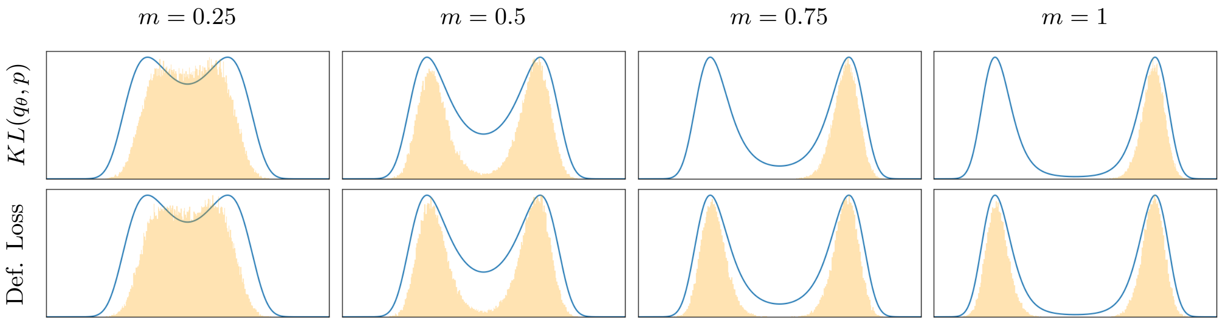

Figure 9: Mode collapse of the reverse KL-divergence for higher values of m. The unnormalized density $p(\phi) \propto e^{-V(\phi)}$ (blue curve), plotted against the flattened samples ϕ_i (orange histogram).

	$m = 0.25$			$m = 0.50$		
	$H(M, X)$ ↓	Rev. KL ↓	Rev. ESS ↑	$H(M, X)$ ↓	Rev. KL ↓	Rev. ESS ↑
$KL(q_\theta, p)$	$1.074 \pm .022$	$-9.777 \pm .001$	$0.984 \pm .001$	$0.879 \pm .007$	$-18.70 \pm .005$	$0.959 \pm .004$
Def. Loss	$1.063 \pm .021$	$-9.781 \pm .001$	$0.996 \pm .000$	$0.864 \pm .014$	$-18.71 \pm .002$	$0.987 \pm .002$
	$m = 0.75$			$m = 1.00$		
	$H(M, X)$ ↓	Rev. KL ↓	Rev. ESS ↑	$H(M, X)$ ↓	Rev. KL ↓	Rev. ESS ↑
$KL(q_\theta, p)$	$16.59 \pm .169$	$-36.18 \pm .003$	$0.958 \pm .006$	$20.13 \pm .088$	$-62.82 \pm .003$	$0.961 \pm .011$
Def. Loss	$\mathbf{0.755} \pm .018$	$\mathbf{-36.88} \pm .005$	$0.967 \pm .017$	$\mathbf{0.695} \pm .018$	$\mathbf{-63.51} \pm .004$	$0.963 \pm .011$

Table 2: Results of training the same flow with two different objectives: the reverse KL-divergence and the deformation loss with the trainable interpolation. Mean and standard deviation values over 3 seeds are reported.

Sensitivity to the dimensionality N. In this section we fix $m = 0.5$ and train models in different dimensionalities $N \in \{4, 8, 16, 32, 64\}$. For each N we trained the same model for the same number of steps. Figure 10 shows the evolution of the *ESS* and $H(M, X)$ values during the training process. To tackle the dropping performance at increasing values of N, improvements of the models are necessary, which we discuss in detail in the next section.

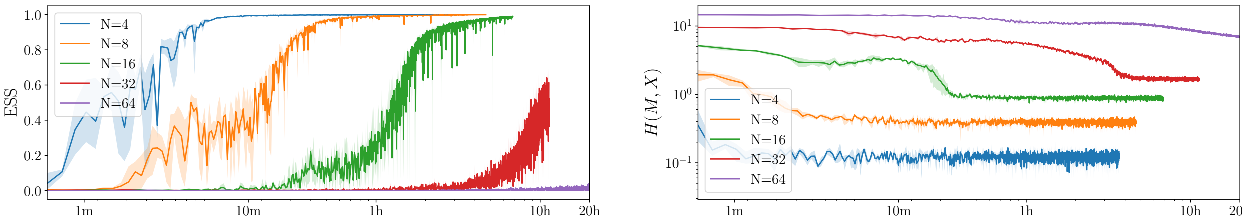

Figure 10: Sensitivity of the ESS and $H(M, X)$ values to the dimensionality of the problem. Mean and standard deviation values over 3 seeds are plotted.

Implementation and training details

Every trainable object in our experiments is parametrized by a weighted sum of MLPs. The weighting is done by evenly spaced RBF time-kernels, one for each model. Importantly, our architecture is completely oblivious to the $\mathbb{Z}_2 \ltimes C_n$-symmetry of the the quantum mechanical particle in a double-well potential. We rely on automatic differentiation to compute all derivatives. We leave the exploitation of symmetries and the use of architectures with analytic expressions for the divergence of V_t (Köhler et al., 2020; Gerdes et al., 2022), as well as for ∇f_t and $\partial_t f_t$, for future work. The choices of hyperparameters are given in Table 3. Everything was implemented in JAX (Bradbury et al., 2018) and executed on one of eight A100 GPUs.

	Gaussian Mixtures		ϕ^4 on a circle
	Eq. 2	Eq. 11	
number of RBF-kernels in time	4	4	8
hidden layers per model	2	2	3
neurons per hidden layer	64	64	128
activation function	swish	swish	swish
base distribution	$\mathcal{N}(0,1)$	$\mathcal{N}(0,2)$	$\propto e^{-\lvert x\rvert^4}$
bath size during training	256	256	256
batch size during evaluation	4096	4096	4096
number of train steps	10^4	10^4	1.5×10^4
initial learning rate	3×10^{-3}	3×10^{-3}	$\times 10^{-3}$
number of integration steps	50	50	50
deformation loss	$\lvert\mathcal{E}\rvert + \lvert\mathcal{E}\rvert^2$	$\lvert\mathcal{E}\rvert + \lvert\mathcal{E}\rvert^2$	$\lvert\mathcal{E}\rvert + \lvert\mathcal{E}\rvert^2$

Table 3: Hyperparameters and design choices for our experiments. The learning rate was initialized to the value shown in the table and annealed to 0 following a cosine schedule.

On the choice of the function L. The definition of the the deformation loss in Eq. 9 involves a somewhat arbitrary choice of the function L. We compared the functions $\{|\mathcal{E}| \mapsto |\mathcal{E}|, |\mathcal{E}| \mapsto |\mathcal{E}|^2, |\mathcal{E}| \mapsto |\mathcal{E}| + |\mathcal{E}|^2, |\mathcal{E}| \mapsto \log(1 + |\mathcal{E}|)\}$ and ended up using $|\mathcal{E}| + |\mathcal{E}|^2$ in all of our experiments, as it empirically outperformed the other choices. The intuition is that the L^2 norm provides good gradients when the error is large, while the L^1 norm provides good gradients when the error is small and therefore their sum is expected to be superior to both of them.

Computational costs. To optimize the "baseline" reverse KL objective one needs a parametrization of V_t and to integrate $\nabla \cdot V_t$ along the trajectories. In addition to this, the deformation loss also needs to parametrize C_t and f_t and to integrate an expression depending on $\nabla \cdot V_t, \partial_t f, \nabla f$ and C_t. This means that the same number of training steps are more expensive computationally when using the deformation loss.

When using the deformation loss instead of the reverse KL-divergence the experiments lasted ~ 2.1-times longer on the Gaussian mixtures and ~ 1.85-times longer on the quantum particle in a double-well potential. In the case of the prescribed linear interpolation on the Gaussian targets, the training lasted ~ 1.6 times longer than with the reverse KL.

5 Summary and Closing remarks

We introduced an alternative training objective of continuous normalizing flows that uses an interpolation of energy functions. The work was motivated by denoising diffusion (Sohl-Dickstein et al., 2015; Ho et al., 2020) and score-base models (Song & Ermon, 2019). It also bears similarities to more recent works (Lipman et al., 2022; Albergo & Vanden-Eijnden, 2022; Liu et al., 2022; Neklyudov et al., 2022) that also learn more flexible interpolations between the data and the base distibutions. We've demonstrated empirically that the proposed objective outperforms the reverse KL-divergence when the target density has multiple modes.

Two reasons for mode collapse. There can be, at least, two different reasons for mode collapse when training with the reverse KL-divergence. First, if one of the modes is so far away from the base distribution that it never gets visited by the flow. Second, if a mode is visited during training by trajectories of the flow, but the flow still ignores it and fits the remaining modes of the density. It is important to distinguish these two scenarios as our proposed technique can help with the second kind, but not the first one. This is also the reason why the standard deviation of the base density is important. When the base has a higher variance, a larger fraction of the space gets visited.

Reframing the method as a PINN. Our work naturally fits into the framework of Physics-informed neural networks (Raissi et al., 2019). What this work calls the pointwise deformation error, would be called the residual to the deformation equation in the PINN literature. The core idea is essentially the same: the optmization of a neural network to satisfy a PDE pointwise.

References

Ryan Abbott, Michael S. Albergo, Denis Boyda, Kyle Cranmer, Daniel C. Hackett, Gurtej Kanwar, Sébastien Racanière, Danilo J. Rezende, Fernando Romero-López, Phiala E. Shanahan, Betsy Tian, and Julian M. Urban. Gauge-equivariant flow models for sampling in lattice field theories with pseudofermions, 2022. URL `https://arxiv.org/abs/2207.08945`.

Michael S Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. *arXiv preprint arXiv:2209.15571*, 2022.

Michael S. Albergo, Denis Boyda, Daniel C. Hackett, Gurtej Kanwar, Kyle Cranmer, Sébastien Racanière, Danilo Jimenez Rezende, and Phiala E. Shanahan. Introduction to normalizing flows for lattice field theory, 2021a. URL `https://arxiv.org/abs/2101.08176`.

Michael S. Albergo, Gurtej Kanwar, Sé bastien Racanière, Danilo J. Rezende, Julian M. Urban, Denis Boyda, Kyle Cranmer, Daniel C. Hackett, and Phiala E. Shanahan. Flow-based sampling for fermionic lattice field theories. *Physical Review D*, 104(11), dec 2021b. doi: 10.1103/physrevd.104.114507. URL `https://doi.org/10.1103%2Fphysrevd.104.114507`.

Michael S. Albergo, Denis Boyda, Kyle Cranmer, Daniel C. Hackett, Gurtej Kanwar, Sébastien Racanière, Danilo J. Rezende, Fernando Romero-López, Phiala E. Shanahan, and Julian M. Urban. Flow-based sampling in the lattice schwinger model at criticality, 2022. URL `https://arxiv.org/abs/2202.11712`.

Denis Boyda, Gurtej Kanwar, Sébastien Racanière, Danilo Jimenez Rezende, Michael S. Albergo, Kyle Cranmer, Daniel C. Hackett, and Phiala E. Shanahan. Sampling using SU(n) gauge equivariant flows. *Phys. Rev. D*, 103:074504, Apr 2021. doi: 10.1103/PhysRevD.103.074504. URL `https://link.aps.org/doi/10.1103/PhysRevD.103.074504`.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/google/jax`.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations, 2018. URL `https://arxiv.org/abs/1806.07366`.

Pim de Haan, Corrado Rainone, Miranda C. N. Cheng, and Roberto Bondesan. Scaling up machine learning for quantum field theory with equivariant continuous flows, 2021. URL `https://arxiv.org/abs/2110.02673`.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

Mathis Gerdes, Pim de Haan, Corrado Rainone, Roberto Bondesan, and Miranda C. N. Cheng. Learning lattice quantum field theories with equivariant continuous flows, 2022. URL `https://arxiv.org/abs/2207.00283`.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Chin-Wei Huang, Laurent Dinh, and Aaron Courville. Augmented normalizing flows: Bridging the gap between generative flows and latent variable models. *arXiv preprint arXiv:2002.07101*, 2020.

Jonas Köhler, Leon Klein, and Frank Noé. Equivariant flows: Exact likelihood generative learning for symmetric densities, 2020. URL `https://arxiv.org/abs/2006.02425`.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

Bálint Máté and François Fleuret. Deformations of boltzmann distributions. *arXiv preprint arXiv:2210.13772*, 2022.

Bálint Máté, Samuel Klein, Tobias Golling, and François Fleuret. Flowification: Everything is a normalizing flow. *arXiv preprint arXiv:2205.15209*, 2022.

Laurence Illing Midgley, Vincent Stimper, Gregor NC Simm, Bernhard Schölkopf, and José Miguel Hernández-Lobato. Flow annealed importance sampling bootstrap. *arXiv preprint arXiv:2208.01893*, 2022.

Radford M. Neal. Annealed importance sampling, 1998. URL `https://arxiv.org/abs/physics/9803008`.

Kirill Neklyudov, Daniel Severo, and Alireza Makhzani. Action matching: A variational method for learning stochastic dynamics from samples. *arXiv preprint arXiv:2210.06662*, 2022.

Kim A. Nicoli, Shinichi Nakajima, Nils Strodthoff, Wojciech Samek, Klaus-Robert Müller, and Pan Kessel. Asymptotically unbiased estimation of physical observables with neural samplers. *Physical Review E*, 101(2), feb 2020. doi: 10.1103/physreve.101.023304. URL `https://doi.org/10.1103%2Fphysreve.101.023304`.

Kim A Nicoli, Christopher J Anders, Lena Funcke, Tobias Hartung, Karl Jansen, Pan Kessel, Shinichi Nakajima, and Paolo Stornati. Estimation of thermodynamic observables in lattice field theories with deep generative models. *Physical review letters*, 126(3):032001, 2021.

Didrik Nielsen, Priyank Jaini, Emiel Hoogeboom, Ole Winther, and Max Welling. Survae flows: Surjections to bridge the gap between vaes and flows. *Advances in Neural Information Processing Systems*, 33:12685–12696, 2020.

Frank Noé, Simon Olsson, Jonas Köhler, and Hao Wu. Boltzmann generators – sampling equilibrium states of many-body systems with deep learning, 2018. URL `https://arxiv.org/abs/1812.01729`.

David Pfau and Danilo Rezende. Integrable nonparametric flows. *arXiv preprint arXiv:2012.02035*, 2020.

M. Raissi, P. Perdikaris, and G.E. Karniadakis. 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: https://doi.org/10.1016/j.jcp.2018.10.045. URL `https://www.sciencedirect.com/science/article/pii/S0021999118307125`.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pp. 2256–2265. PMLR, 2015.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

Esteban G Tabak and Cristina V Turner. A family of nonparametric density estimation algorithms. *Communications on Pure and Applied Mathematics*, 66(2):145–164, 2013.

Lorenz Vaitl, Kim A Nicoli, Shinichi Nakajima, and Pan Kessel. Gradients should stay on path: better estimators of the reverse- and forward kl divergence for normalizing flows. *Machine Learning: Science and Technology*, 3(4):045006, oct 2022. doi: 10.1088/2632-2153/ac9455. URL `https://dx.doi.org/10.1088/2632-2153/ac9455`.