Matching Normalizing Flows and Probability Paths on Manifolds

Heli Ben-Hamu [*1] **Samuel Cohen** [*2] **Joey Bose** [2] **Brandon Amos** [2] **Aditya Grover** [2]
Maximilian Nickel [2] **Ricky T. Q. Chen** [2] **Yaron Lipman** [1 2]

Abstract

Continuous Normalizing Flows (CNFs) are a class of generative models that transform a prior distribution to a model distribution by solving an ordinary differential equation (ODE). We propose to train CNFs on manifolds by minimizing *probability path divergence* (PPD), a novel family of divergences between the probability density path generated by the CNF and a target probability density path. PPD is formulated using a logarithmic mass conservation formula which is a linear first order partial differential equation relating the log target probabilities and the CNF's defining vector field. PPD has several key benefits over existing methods: it sidesteps the need to solve an ODE per iteration, readily applies to manifold data, scales to high dimensions, and is compatible with a large family of target paths interpolating pure noise and data in finite time. Theoretically, PPD is shown to bound classical probability divergences. Empirically, we show that CNFs learned by minimizing PPD achieve state-of-the-art results in likelihoods and sample quality on existing low-dimensional manifold benchmarks, and is the first example of a generative model to scale to moderately high dimensional manifolds.

1. Introduction

One of the core domains of machine learning research are density estimation and generative modeling, which view data from a probabilistic perspective. The deep-learning revolution fostered a significant advancement in the field, leading to the emergence of powerful generative models for images, language, audio, and other data types represented in Euclidean spaces. While early literature primarily focused on Euclidean data, the need to model data in non-Euclidean spaces arises in many scientific fields. For instance, occurrences of natural phenomena on earth can be modeled as a distribution on a sphere (Mathieu & Nickel, 2020), protein structure prediction requires angle predictions (Mardia et al., 2008) and motion and position of robots can be modeled with a product of Euclidean spaces and spheres. Therefore, constructing generative models over manifolds is an important problem with many potential applications.

A generative model can be described as a function ϕ that transforms a simple probability distribution (the prior or base) to a more complicated one (the model) so to best represent some empirical set of data observations. Among the large toolkit of deep generative models, innate candidates for designing generative models on manifolds are Normalizing Flows (NFs) (Rezende & Mohamed, 2015) and Continuous Normalizing Flows (CNFs) (Chen et al., 2018). In these approaches ϕ is a diffeomorphism, i.e., a smooth bijection with a smooth inverse. Therefore, the model density can be expressed in terms of the prior density and the determinant of the Jacobian of ϕ, also known as the change of variable formula, which can be naturally adapted to the manifold case. Recently, (Rezende et al., 2020; Bose et al., 2020) devised NF models for sphere, tori and hyperbolic spaces. In a parallel line of works, (Mathieu & Nickel, 2020; Lou et al., 2020; Falorsi & Forré, 2020) developed CNFs over Riemannian manifolds.

Like Euclidean NF models, manifold NFs suffer from limited expressive power due to the representation of ϕ as a composition of a restricted set of invertible transformations. On the other hand, CNFs model diffeomorphisms as flows along parametric tangent vector fields, v, represented as neural networks, lifting the architectural restriction and allowing maximal expressive power. Nonetheless, training a CNF by minimizing negative log-likelihoods, or equivalently, the KL-divergence of the data and model densities, requires log model densities evaluated at observation points. Computing the log model densities entails solving an ordinary differential equation (ODE) during training, which results in a substantial time and memory burden, as well as introduces an extra challenge when the data lies on a manifold. Rozen et al. (2021) suggested a different parametrization of CNFs via the divergence of unrestricted vector fields, where both training and computing model probabilities do

[*]Equal contribution [1]Weizmann Institute of Science [2]Meta Research. Correspondence to: Heli Ben-Hamu <heli.benhamu@weizmann.ac.il>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

not require solving an ODE. However, scaling this method to even moderately high dimensions is challenging since it is formulated with a density function rather than log density, which can cause numerical issues as density values decrease exponentially with dimension.

This work aims to alleviate some of the limitations of previous approaches by introducing the Probability Path Divergences (PPD), a new type of divergence defined between an arbitrary target probability path, p, and the probability path generated by the CNF, q. To define the PPD we first introduce the Logarithmic Mass Conservation (LMC) formula, a Partial Differential Equation (PDE) that couples $\log q$ and the CNF's vector field. Then, the PPD is defined as the extent to which $\log p$ and the CNF's vector field fail to satisfy the LMC. PPD has the following desirable properties: (i) It is a proper divergence in the sense that it is non-negative, and zero iff $p \equiv q$. (ii) It does not require evaluating q during training; it is defined solely in terms of the parametric vector field v, its first order derivatives, and the target path's log density, $\log p$. This provides a speed up of $1-2$ orders of magnitude in evaluating the PPD and its derivatives, compared to, e.g., log likelihood. (iii) It is readily applicable to manifolds and higher dimensional data. (iv) The PPD has a single parameter $\ell \geq 1$. PPD with $\ell = 1$ upper bounds the total-variation divergence comparing p and q at arbitrary times; PPD with $1 < \ell < \infty$ bounds their α-divergence; and PPD with $\ell = \infty$ bounds their reversed KL-divergence.

We call the minimization problem of the PPD between a target path p and a CNF density q CNF Matching (CNFM), and use it to train CNFs. The main design choice in CNFM is the target path p. The requirements from p are: that it transforms a simple prior (pure noise) to an approximation of the unknown data distribution; that samples can be drawn from each p_t, where p_t represents the density at time t; and that we can compute or approximate the derivatives of $\log p_t$. Any p satisfying these requirements can be used to train a CNF in the CNF Matching framework. Other methods that try to fit generated probability density path to a target one are Score and Diffusion based methods (Song & Ermon, 2019; Ho et al., 2020; Song et al., 2020). However, these methods require target paths that are generated by Stochatsic Differential Equations (SDEs) or known diffusion processes which limits their applicability on manifolds. We elaborate this discussion in Section 4.2, after introducing our method.

We test our framework on several low and moderately high dimensional manifold data including Euclidean spaces, spheres/hyperspheres, and product of spheres, demonstrating state-of-the-art sample quality and likelihoods in standard low-dimensional manifold datasets. We demonstrate that CNFM is considerably faster to optimize than state of the art CNF training algorithm, allowing to scale CNF training to considerably larger network architectures. Lastly, we demonstrate that CNFM can train CNFs on moderately high dimensional manifolds, in contrast to previous methods of generative modeling on manifold that mostly worked with low dimensional manifolds.

2. Preliminaries

Let $\mathcal{M}$ be a d-dimensional smooth Riemannian manifold with a metric g and induced volume form dV, the volume of $\mathcal{M}$ is $|\mathcal{M}| = \int_{\mathcal{M}} dV_x$. We consider strictly positive, smooth probability densities over $\mathcal{M}$, $\mu : \mathcal{M} \to \mathbb{R}_{>0}$, satisfying $\int_{\mathcal{M}} \mu(x) dV_x = 1$. The tangent space at point $x \in \mathcal{M}$ is denoted $T_x\mathcal{M}$; the tangent bundle, which is the disjoint union of all tangent spaces of $\mathcal{M}$ is denoted $T\mathcal{M}$. The metric g defines an inner product for pairs of vectors $\xi, \eta \in T_x\mathcal{M}$ denoted by $\langle \xi, \eta \rangle$; a norm of a tangent vector is defined by $|\xi| = \langle \xi, \xi \rangle^{1/2}$. The Riemannian gradient of a smooth function $f : \mathcal{M} \to \mathbb{R}$ is denoted $\nabla f(x) \in T_x\mathcal{M}$. A time-dependent vector field $v(t, x)$ is a smooth function $v : [0,1] \times \mathcal{M} \to T\mathcal{M}$ such that $v(t,x) \in T_x\mathcal{M}$ for all $t \in [0,1]$ and $x \in \mathcal{M}$. We denote the collection of bounded time dependent smooth vector fields over $\mathcal{M}$ by $\mathfrak{X}(\mathcal{M})$; by bounded we mean that for each $v \in \mathfrak{X}(\mathcal{M})$ there exists a constant $M > 0$ so that $|v(t,x)| \leq M$ for all $x \in \mathcal{M}$, $t \in [0,1]$. The Riemannian divergence (w.r.t. x) of a smooth vector field $v \in \mathfrak{X}(\mathcal{M})$ is denoted $\mathrm{div}(v)$. We denote by $\exp_x : T_x\mathcal{M} \to \mathcal{M}$, and $\log_x : \mathcal{M} \to T_x\mathcal{M}$ the Riemannian exponential and logarithmic maps. Note these should not be confused with the standard $\exp, \log$ that are written without subscript.

Given a time dependent vector field $v \in \mathfrak{X}(\mathcal{M})$, a one parameter diffeomorphism family $\phi_t : \mathcal{M} \to \mathcal{M}$ can be defined via the Ordinary Differential Equation (ODE):

$$\begin{cases} \frac{d}{dt}\phi_t(x) = v(t, \phi_t(x)) \\ \phi_0(x) = x \end{cases} \tag{1}$$

In the context of generative models, the diffeomorphism ϕ_t is called a Continuous Normalizing Flow (CNF) (Chen et al., 2018; Mathieu & Nickel, 2020; Lou et al., 2020; Falorsi & Forré, 2020; Rozen et al., 2021) and is used to push-forward or pull-back probability densities. An event $A \subset \mathcal{M}$ is pushed forward by ϕ_t to the event $\phi_t(A)$, and pulled back to $\phi_t^{-1}(A)$. Given a probability density η over $\mathcal{M}$ its pushed forward density is denoted $\phi_{t*}\eta$, and its pulled back density by ϕ_t is denoted $\phi_t^*\eta$. Let $\mathfrak{P}(\mathcal{M})$ denote all *probability paths* on $\mathcal{M}$, that is functions $p : [0,1] \times \mathcal{M} \to \mathbb{R}_{>0}$, smooth in t and satisfying $\int_{\mathcal{M}} p(t,x) dV_x = 1$.

Definition 1. *We say that a CNF ϕ_t* generates *a probability density path $q \in \mathfrak{P}(\mathcal{M})$ if for all $t \in [0,1]$*

$$q_t = \phi_{t*} q_0, \text{ or equivalently } \phi_t^* q_t = q_0, \tag{2}$$

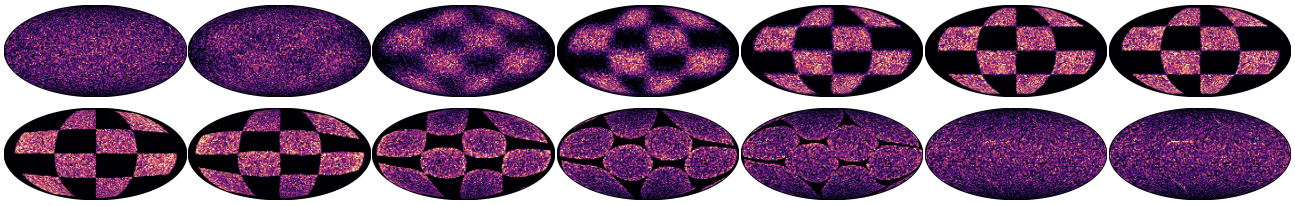

Figure 1. CNFM on a manifold (sphere): the trained CNF ϕ_t is pushing noise $x \sim p_0$ to data $\phi_t(x)$ (top, from left $t = 0$ to right $t = 1$); and the reverse time CNF taking data $x \sim p_{\text{data}}$ to noise $\phi_{1-t}(x)$ (bottom).

3. Matching CNF and target probability

We start by considering a *target* probability density path $p \in \mathfrak{P}(\mathcal{M})$. We will use the notation p_t to denote the density at time t, namely, $p_t = p(t, \cdot)$. In a typical target path p, p_0 is some simple prior distribution, e.g., a distribution representing pure noise, and p_1 approximates the unknown data distribution, denoted p_{data} and is practically approximated by some empirical set of samples.

Our goal is to match p and the density path $q \in \mathfrak{P}(\mathcal{M})$, generated by a CNF ϕ_t from the prior p_0. The CNF ϕ_t is defined by equation 1 via a learnable time dependent vector field $v_\theta \in \mathfrak{X}(\mathcal{M})$, with parameters $\theta \in \mathbb{R}^p$. In more detail, we define the CNF Matching (CNFM) problem as the following optimization problem:

$$\min_\theta \quad \mathrm{d}(p \,\|\, q) \tag{3a}$$

$$\text{s.t.} \quad q_t = \phi_{t*} p_0, \quad t \in [0, 1] \tag{3b}$$

where d is a probability divergence between probability density *paths*. That is, for density paths $p, q \in \mathfrak{P}(\mathcal{M})$, $\mathrm{d}(p, q) \geq 0$, and $\mathrm{d}(p, q) = 0$ iff $p_t \equiv q_t$ for all $t \in [0, 1]$.

Adapting existing CNF approaches to optimize equation 3 would require evaluating q_t, which is provided only through solutions to an ODE (see also the discussion in Section 4.1), and will therefore introduce a substantial computational challenge. Instead, we construct a novel divergence d, called the Probability Path Divergence (PPD), that does not require sampling of q or enforcing equation 3b explicitly, and therefore sidesteps the need for solving ODE during training. Furthermore, we will show that PPD bounds standard probability divergences such as total variation, α, and reverse KL. Figure 1 depicts an example of a CNF, ϕ_t, trained with CNFM and PPD using a target path p that is interpolating between uniform and checkerboard data over the sphere. In the top row we depict random uniform samples over the sphere $x \sim p_0$ (left) pushed by the CNF, i.e., $\phi_t(x)$, for several times $t \in [0, 1]$, reaching the desired checkerboard distribution at $t = 1$ (right). The bottom row shows the CNF pulling, i.e., $\phi_{1-t}(x)$, data samples $x \sim p_{\text{data}}$ (left), reaching a uniform distribution at time $t = 1$ (right).

3.1. Logarithmic Mass Conservation

As a first step in constructing the PPD we derive a Partial Differential Equation (PDE) involving the log density path $\log p$ and a vector field v, such that it is satisfied iff the CNF ϕ_t, defined by v, generates p. We name this equation the Logarithmic Mass Conservation (LMC) formula.

Theorem 1. *Consider a CNF $\phi_t : \mathcal{M} \to \mathcal{M}$ defined by a smooth, time dependent vector field $v \in \mathfrak{X}(\mathcal{M})$ as in equation 1, and a probability density path $p \in \mathfrak{P}(\mathcal{M})$. Then p is generated by ϕ_t, i.e.,*

$$p_t = \phi_{t*} p_0, \quad \forall t \in [0, 1] \tag{4}$$

if and only if the LMC formula holds over $[0, 1] \times \mathcal{M}$:

$$\partial_t \log p_t + \langle \nabla \log p_t, v \rangle + \operatorname{div}(v) = 0 \tag{5}$$

The LMC formula can be proved with the aid of the mass conservation formula, also known as the continuity equation and equivalent to equation 5 (Villani, 2009):

$$\partial_t p_t + \operatorname{div}(p_t v) = 0, \tag{6}$$

where div denotes the divergence operator over the manifold $\mathcal{M}$. We assumed $p > 0$ and therefore dividing both sides by p_t leads to

$$\frac{\partial_t p_t}{p_t} + \frac{\langle \nabla p_t, v \rangle + p_t \operatorname{div}(v)}{p_t} = 0,$$

where we also used the fact that $\operatorname{div}(fv) = \langle \nabla f, v \rangle + f \operatorname{div}(v)$. Finally noting that $\partial_t \log p_t = \frac{\partial_t p_t}{p_t}$, and $\nabla_x \log p_t = \frac{\nabla_x p_t}{p_t}$ we get that equation 5 is equivalent to equation 6. See Appendix A for more details.

The benefit of using the LMC formula over the standard mass conservation formula is that it is formulated directly in terms of the log probability $\log p_t$, which reduces numerical issues for high dimensions.

3.2. Probability path divergence

Plugging a fixed target path $p \in \mathfrak{P}(\mathcal{M})$ in the LMC formula (equation 5) provides a necessary and sufficient condition for v to generate p via a CNF. Motivated by this observation, we define a family of probability path divergences (PPD), parameterized by an integer $\ell \geq 1$, comparing $p, q \in \mathfrak{P}(\mathcal{M})$ where $q_t = \phi_{t*} p_0$:

$$\mathrm{d}_\ell(p \,\|\, q) = \mathbb{E}_{t,x\sim p_t} \left| \partial_t \log p_t + \langle \nabla \log p_t, v \rangle + \mathrm{div}(v) \right|^\ell \quad (7)$$

where t is distributed over $[0,1]$, e.g., uniform $t \sim \mathcal{U}[0,1]$. $\mathrm{d}_\ell(p \,\|\, q) \geq 0$ by construction, and Theorem 1 implies that $\mathrm{d}_\ell(p \,\|\, q) = 0$ iff $p_t \equiv q_t$ for all $t \in [0,1]$. Using this path divergence in the CNFM problem (equation 3) we arrive to the following instantiation:

$$\min_\theta \mathbb{E}_{t,x\sim p_t} \left| \partial_t \log p_t + \langle \nabla \log p_t, v_\theta \rangle + \mathrm{div}(v_\theta) \right|^\ell \quad (8)$$

where v_θ is the learnable vector field defining the CNF ϕ_t generating q_t. Importantly, evaluating the PPD $\mathrm{d}_\ell(p \,\|\, q)$ and its derivatives with respect to θ does not require access to q and ϕ_t, and therefore sidesteps solving the ODE in equation 1 during training.

The following Theorem relates the path divergence d_ℓ to standard divergences of probability densities. We consider f-divergences (Ali & Silvey, 1966; Csiszár, 1967) of two probability densities μ, ν defined by

$$D_f(\mu \,\|\, \nu) = \int_{\mathcal{M}} f\left(\frac{\mu(x)}{\nu(x)}\right) \nu(x) dV_x \quad (9)$$

where $f : \mathbb{R}_{\geq 0} \to \mathbb{R}$ is a strictly convex function satisfying $f(1) = 0$. f-divergences satisfy the standard statistical divergence properties: $D_f(\mu \,\|\, \nu) \geq 0$, and $D_f(\mu \,\|\, \nu) = 0$ iff $\mu \equiv \nu$. f-divergences generalize standard divergences such as KL (with the choice $f(t) = t \log t$), reverse KL ($f(t) = -\log t$), total variation ($f(t) = |t-1|$), and α-divergences ($f(t) = 1 - t^\alpha$ with $\alpha \neq 1, 0$). We prove:

Theorem 2. *Consider paths $p, q \in \mathfrak{P}(\mathcal{M})$ where q is generated by a CNF $\phi_t : \mathcal{M} \to \mathcal{M}$, and $q_0 = p_0$. Then for all $T \in [0,1]$*

$$\mathrm{d}_\ell(p \,\|\, q)^{\frac{1}{\ell}} \geq D_f(p_T \,\|\, q_T) \quad (10)$$

where

$$f(t) = \begin{cases} |t-1| & \ell = 1 \quad \textit{(total variation)} \\ \ell\left(1 - t^{\frac{1}{\ell}}\right) & 1 < \ell < \infty \quad (\alpha) \\ -\log t & \ell = \infty \quad \textit{(reverse KL)} \end{cases}$$

Theorem 2 shows that the path divergence d_ℓ bounds the respective f-divergences of p_T and q_T for all times $T \in [0,1]$. Figure 2 visualizes four instances of f corresponding to different choices of ℓ. Note, that with the exception of $\ell = 1$, all f are differentiable and have the same derivative at 1, which means they have similar value and derivatives when evaluating the divergence of nearby probability densities. As $\ell \to \infty$ we can see the f functions gets close to the $-\log t$ limit.

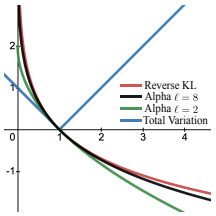

Figure 2. f instances, see Theorem 2.

Specifically, in the $\ell = \infty$ case of Theorem 2, we mean that the inequality equation 10 holds in the limit as $\ell \to \infty$, or more precisely,

$$\liminf_{\ell \to \infty} \mathrm{d}_\ell(p \,\|\, q)^{1/\ell} \geq D_f(p_T \,\|\, q_T),$$

where we also assume that $D_f(p_T \,\|\, q_T) < \infty$. To prove this theorem we will use the following lemma, proved in Appendix B.

Lemma 1. *Consider paths $p, q \in \mathfrak{P}(\mathcal{M})$ where q is generated by a CNF $\phi_t : \mathcal{M} \to \mathcal{M}$, and $q_0 = p_0$. Then the following holds:*

$$\mathrm{d}_\ell(p \,\|\, q) = \mathbb{E}_{x\sim p_0} \int_0^1 \frac{p_t(\phi_t(x))}{q_t(\phi_t(x))} \left| \partial_t \left[\log \frac{p_t(\phi_t(x))}{q_t(\phi_t(x))} \right] \right|^\ell dt$$

We now use Lemma 1 to prove each case of Theorem 2:

$\ell = 1$ case. For $\ell = 1$, Lemma 1 provides the following form for d_1:

$$\mathrm{d}_1(p \,\|\, q) = \mathbb{E}_{x\sim p_0} \int_0^1 \left| \partial_t \left[\frac{p_t(\phi_t(x))}{q_t(\phi_t(x))} \right] \right| dt \quad (11)$$

which shows that for $\ell = 1$ the path divergence is equivalent to the Total Variation norm of the density ratio p_t/q_t along trajectories of the flow. Second, Jensen's inequality with the convex function $|\cdot|$ provides for every $T \in [0,1]$

$$\begin{aligned} \mathrm{d}_1(p \,\|\, q) &\geq \mathbb{E}_{x\sim p_0} \int_0^T \left| \partial_t \left[\frac{p_t(\phi_t(x))}{q_t(\phi_t(x))} \right] \right| dt \\ &\geq \mathbb{E}_{x\sim p_0} \left| \frac{p_T(\phi_T(x))}{q_T(\phi_T(x))} - 1 \right| = \mathbb{E}_{x\sim q_T} \left| \frac{p_T(x)}{q_T(x)} - 1 \right| \\ &= D_f(p_T, q_T) \end{aligned}$$

where the first inequality is due to the fact that we integrate over the smaller interval $[0,T]$, in the first equality we used the fact that $\phi_T(x) \sim q_T$ if $x \sim p_0$, and in the last equality we took $f(t) = |t-1|$.

$1 < \ell < \infty$ **case.** Lemma 1 again with Jensen's inequality of the convex function $|\cdot|^\ell$ provides

$$\begin{aligned}
\mathrm{d}_\ell(p \,\|\, q)^{\frac{1}{\ell}} &\geq \left| \mathbb{E}_{x\sim p_0} \int_0^T \left[\frac{p_t(\phi_t(x))}{q_t(\phi_t(x))}\right]^{\frac{1}{\ell}} \partial_t \left[\log \frac{p_t(\phi_t(x))}{q_t(\phi_t(x))}\right] dt \right| \\
&= \left| \mathbb{E}_{x\sim p_0} \int_0^T \ell \partial_t \left[\frac{p_t(\phi_t(x))}{q_t(\phi_t(x))}\right]^{\frac{1}{\ell}} dt \right| \\
&= \left| \mathbb{E}_{x\sim p_0} \ell \left(\left[\frac{p_T(\phi_T(x))}{q_T(\phi_T(x))}\right]^{\frac{1}{\ell}} - 1 \right) \right| \\
&= \left| \mathbb{E}_{x\sim q_T} \ell \left(\left[\frac{p_T(x)}{q_T(x)}\right]^{\frac{1}{\ell}} - 1 \right) \right| \qquad (12) \\
&= D_f(p_T \,\|\, q_T)
\end{aligned}$$

with $f(t) = \ell(1 - t^{\frac{1}{\ell}})$.

$\ell = \infty$ **case.** First, we note that for any $t > 0$, $\ell(1-t^{\frac{1}{\ell}}) \nearrow -\log(t)$, that is, $\ell(1 - t^{\frac{1}{\ell}})$ is monotonically increasing and converging to $-\log(t)$ as $\ell \to \infty$ (see Appendix C for a proof). Next, consider equation 12 and move to the limit $\ell \to \infty$:

$$\begin{aligned}
\liminf_{\ell\to\infty} \mathrm{d}_\ell(p \,\|\, q)^{\frac{1}{\ell}} &\geq \lim_{\ell\to\infty} \left| \mathbb{E}_{x\sim q_T} \ell \left(1 - \left[\frac{p_T(x)}{q_T(x)}\right]^{\frac{1}{\ell}} \right) \right| \\
&= -\mathbb{E}_{x\sim q_T} \log \frac{p_T(x)}{q_T(x)} \qquad (13) \\
&= D_f(p_T \,\|\, q_T)
\end{aligned}$$

with $f(t) = -\log(t)$. The previous to last equality (integral and limit switch) is justified in Appendix C; the minus sign is due to the fact that $D_f(p_T \,\|\, q_T) \geq 0$.

3.3. Target paths

The last ingredient needed for defining the probability path divergence (equation 7) is the target path $p \in \mathfrak{P}(\mathcal{M})$. In our framework, p should be defined satisfying the following requirement:

(i) p_0 is pure noise, e.g., a standard Gaussian or uniform.

(ii) p_1 approximates the unknown data distribution p_{data}.

(iii) We have an efficient generation procedure for $x \sim p_t$.

(iv) We have an approximation procedure for the time (∂_t) and space (∂_x) derivatives of $\log p_t(x)$.

Note that these requirements do not mean we know of an SDE, generating random variables distributed as p_t, nor a PDE (Fokker-Planck) with p_t as its solution. In fact, below we construct paths for which an SDE/PDE characterization is not known. In that context the target paths we consider are general; see discussion in Section 4.2.

In the following we construct target paths $p \in \mathfrak{P}(\mathcal{M})$ for several manifolds of interest. At the base of our construction is a kernel $p_\tau(x|y)$, namely a probability density in $x \in \mathcal{M}$, centered at $y \in \mathcal{M}$, with scale $\tau > 0$. We define our (ideal) target path $p \in \mathfrak{P}(\mathcal{M})$ by

$$p_t(x) = \int_{\mathcal{M}} p_\tau(x \,|\, \gamma_t(y)) p_{\mathrm{data}}(y) dV_y \qquad (14)$$

where $\tau = \tau(t)$, $t \in [0, 1]$, is a time-dependent scale function, and $\gamma : [0, 1] \times \mathcal{M} \to \mathcal{M}$ is some differentiable in t map. In practice we don't know p_{data}, rather, we have an empirical sample $\{y_i\}_{i=1}^m$, drawn i.i.d. from p_{data}. Therefore we use the following approximation of equation 14

$$p_t(x) = \frac{1}{m} \sum_{i=1}^m p_\tau(x \,|\, \gamma_t(y_i)) \qquad (15)$$

Note, that if we know how to compute or approximate $\log p_\tau(x|\gamma_t(y_i))$ then $\log p_t(x)$, required for the computation of the PPD, has the form

$$\log p_t(x) = \texttt{logsumexp}\{\log p_\tau(x|\gamma_t(y_i))\}_{i=1}^m - \log m$$

Depending on the type of manifold, we consider two basic target path constructions that differ in their prior p_0: *Unimodal*, where the prior probability p_0 is centered around a single designated point in $\mathcal{M}$. Unimodal prior is mainly suitable to non-compact manifolds with infinite volume such as Euclidean or hyperbolic spaces. *Uniform*, where the prior p_0 is the uniform density over $\mathcal{M}$. A uniform prior is suitable to compact manifolds such as spheres.

Unimodal prior. Let $o \in \mathcal{M}$ be some designated point, $\sigma_0, \sigma_1 \geq 0$ initial and target scales. We define p according to equation 15 by making the choices:

$$\gamma_t(y) = \exp_o(t \log_o y) \;\;,\;\; \sigma(t) = \sigma_0^{1-t} \sigma_1^t \qquad (16)$$

where $\tau = \sigma$ is the scaling function, and $\gamma_t(y)$ moves y to the center o along a geodesic (we assume the Riemannian $\exp_o, \log_o$ are defined in a sufficiently large neighborhood of o and $T_o\mathcal{M}$). With these choices, p_t starts with a single mode density p_0, centered at $o \in \mathcal{M}$, and then splits the unit mass, moving each $\frac{1}{m}$ part towards the empirical sample y_i along a geodesic while concentrating the density.

Euclidean. Let us instantiate the unimodal path for the Euclidean space, $\mathcal{M} = \mathbb{R}^d$, with the standard metric $\langle v, u \rangle = v^T u$, where $v, u \in \mathbb{R}^d$ are (always) column vectors. Our kernel in this case is the Gaussian, $p_\sigma(x|y) = \mathcal{N}(x|y, \sigma^2 \boldsymbol{I})$ with mean $y \in \mathbb{R}^d$ and covariance $\sigma^2 \boldsymbol{I}$. Furthermore, for $o \in \mathbb{R}^d$, $\exp_o(t \log_o y_i) = o + t(y_i - o) = (1 - t)o + t y_i$.

Therefore, equation 15 takes the form

$$p_t(x) = \frac{1}{m}\sum_{i=1}^{m}\mathcal{N}(x \,|\, (1-t)o + ty_i, \sigma^2 \boldsymbol{I}) \tag{17}$$

and we take $\sigma_0 = 1$ to represent a standard Gaussian prior, i.e., $\sigma(t) = \sigma_1^t$, and $\sigma_1 > 0$ is the (only) hyper-parameter.

Uniform prior. In this family of paths we consider compact manifolds $\mathcal{M}$ and start from the uniform density p_0. We assume in this case we have a kernel $p_\kappa(x|y)$ such that there exists a finite $\kappa_0 \geq 0$, for which $p_{\kappa_0}(x|y) \equiv |\mathcal{M}|^{-1}$ for all $y \in \mathcal{M}$, i.e., p_{κ_0} represents the uniform density. One way to construct such a kernel on compact submanifolds of $\mathbb{R}^{d+1}$, $\mathcal{M} \subset \mathbb{R}^{d+1}$, is by restricting an Euclidean Gaussian in $\mathbb{R}^{d+1}$ to $\mathcal{M}$; we discuss such a construction on the sphere below. In this case we define the target path using equation 15 again by making the choices

$$\gamma_t(y) = y \;\;,\;\; \kappa(t) = (1-\kappa_0+\kappa_1)^t + \kappa_0 - 1 \tag{18}$$

where $\tau = \kappa$ is the scaling function, and $\gamma_t(y)$ leaves samples at their original location.

Sphere. We instantiate the uniform prior paths to the unit spheres $\mathcal{M} = \mathcal{S}^d \subset \mathbb{R}^{d+1}$ with the induced metric from the Euclidean $\mathbb{R}^{d+1}$. The von Mises-Fisher (vMF) kernel (Mardia, 2014) is:

$$p_\kappa(x|y) = c_d(\kappa)\exp(\kappa x^T y), \tag{19}$$

where $c_d(\kappa)$ is the normalization constant detailed in Appendix D. vMF can be seen as a restricted Gaussian $\exp(-\kappa\|x-y\|_2^2)$ to the unit sphere $x, y \in \mathcal{S}^d$ with the relevant normalization constant. For $\kappa = 0$, $p_0(x|y)$ is uniform over the sphere for all $y \in \mathcal{S}^d$. Hence we take $\kappa_0 = 0$, which leaves $\kappa(t) = (1+\kappa_1)^t - 1$, and $\kappa_1 > 0$ is the (only) hyper-parameter in this case. The target path takes the form

$$p_t(x) = \frac{1}{m}\sum_{i=1}^{m} p_\kappa(x|y_i) \tag{20}$$

Paths on products of manifolds. We conclude the section with generalizing the target path construction to product of manifolds. Let $\mathcal{M} = \mathcal{M}^1 \times \ldots \times \mathcal{M}^N$. Each point $x \in \mathcal{M}$ is represented as a tuple $x = (x^1, \ldots, x^N)$, where $x^j \in \mathcal{M}^j$. For example, in robotics, a robot's state can be represented by the sequence of locations and/or rotations of its joints, i.e., each $\mathcal{M}^j$ is either a sphere ($\mathcal{S}^3$ for 3D rotations represented as quaternions; $\mathcal{S}^1$ for 2D rotations) or an Euclidean space (representing positions). Let p_{τ^j} be a kernel defined in $\mathcal{M}^j$, and γ^j is a deformation of $\mathcal{M}^j$. For example, for $\mathcal{M}^j$ being the Euclidean plane or a sphere we can use the above definitions for kernels p_{τ^j}. Let $\{y_i\}_{i=1}^m \subset \mathcal{M}$ be i.i.d. samples from p_{data} over $\mathcal{M}$. We define the kernel for $\mathcal{M}$ by

$$p_\tau(x|y_i) = \prod_{j=1}^{N} p_{\tau^j}(x^j|\gamma_t^j(y_i^j)) \tag{21}$$

We note that $p_\tau(x|y)$ is a probability density in $x \in \mathcal{M}$, and if $p_{\tau^j}(x^j|y^j)$ is concentrated (as a function of $x^j \in \mathcal{M}^j$) around y^j for all j, then $p_\tau(x|y)$ is concentrated (as a function of $x \in \mathcal{M}$) around y. Lastly, and use equation 15 again to define out target path $p \in \mathfrak{P}(\mathcal{M})$. Further implementation details for the vector field v_θ are in Appendix E.

4. Previous works

4.1. Relations to existing CNF models

The LMC formula (equation 5) is a linear first order PDE in $\log p_t$. Solving it using the method of characteristics (Evans, 1997) provides a simple proof of the Instantaneous Change of Variables Theorem from (Chen et al., 2018) and generalizes it to the manifold setting. Indeed, using the chain rule and the LMC we have

$$\begin{aligned}\partial_t\left[\log q_t(\phi_t)\right] &= \partial_t \log q_t(\phi_t) + \langle \nabla_x \log q_t(\phi_t), v(t,\phi_t)\rangle \\ &= -\operatorname{div} v(t,\phi_t)\end{aligned} \tag{22}$$

where $\partial_t\left[\log q_t(\phi_t)\right]$ denotes the total derivative w.r.t. t. Training a neural ODE by maximizing the likelihood of the data points $x_i \in \mathcal{M}$ entails computing $\log q_t(x_i)$ and its derivatives w.r.t. the parameters of the vector field v_t. Using the characteristic method (Chen et al., 2018; Lou et al., 2020; Mathieu & Nickel, 2020; Falorsi & Forré, 2020) this amounts to solving an ODE for $(\log q_t(\phi_t), \phi_t)$ (equations 1 and 22) and differentiating the solution (which involves another ODE solve). In contrast, minimizing the PPD does not require solving an ODE during training.

Moser Flow (MF) (Rozen et al., 2021) suggests to train a CNF by formulating the model density as $q_1 = p_0 - \operatorname{div}(u)$, where u is time independent vector field over $\mathcal{M}$. Its relation to our method can be seen by making the choice $q_t = (1-t)p_0 + tq_1$, where p_0 and q_1 are prior and model probability densities, respectively. Indeed, plugging this path in the mass conservation equation (equation 6) gives

$$q_1 - p_0 + \operatorname{div}(q_t v) = 0$$

which directly leads to MF by plugging $u = p_t v$ as a time

independent solution to this equation. Although MF also avoids solving an ODE during training and generalizes to manifolds, it incorporates an additional loss term for keeping the model density q_1 positive; this loss term has high variance and does not scale to high dimensions. Furthermore, MF models probabilities rather than log-probabilities, which also hinders modeling high dimensional densities. Lastly, MF models a particular probability path (convex combinations of prior and model), while our framework can match more general paths.

4.2. Relations to diffusion and score based generative models

Another body of related work concerns diffusion-based generative models (Sohl-Dickstein et al., 2015; Ho et al., 2020) and SDE/score-based generative models (Song & Ermon, 2019; Song et al., 2020). Both approaches also use a certain probability density path, called the *forward process*, to train their generative model. The forward process is a (continuous or discrete) time dependent noising scheme converting the data distribution to a simple, easy to sample from, prior distribution. In diffusion models the forward process is defined by a markov chain, whereas for score models it is defined by an SDE. The forward process is used for training the *reverse process* parameterized with a neural network. The reverse process is used to generate samples from the prior distributions.

Training diffusion/score models entails: (i) sampling the forward process at arbitrary times t; this requires either a closed-form solution of the respective diffusion/SDE forward process (especially challenging over manifolds, more on this below), or simulating the process from time $t = 0$ (costly). (ii) Spatial derivatives of the log transition kernel. Where the transition kernel is the probability of sampling a point x at time t from the forward process given an initial point $y \sim p_{\text{data}}$. (iii) Known form of the reverse diffusion/SDE process.

Our CNFM approach, based on the LMC formulation, does not require the probability density path p_t to be a known solution of a particular diffusion or SDE process and the reverse process is trivially obtained by solving the ODE in reversed time with the learned v_θ. This makes the path choice in our approach more flexible compared to diffusion, score and SDE models, which are restricted to probability density paths defined by known diffusion processes (e.g., Gaussian) or SDEs with closed form transition kernels.

This flexibility becomes especially important when the domain we want to learn on is not Euclidean. For example, consider the arguably simplest SDEs, describing Brownian motion over $\mathcal{M}$. The corresponding probability kernel $p_t(x|y)$ is the fundamental solution to the heat equation $\partial_t p = \Delta p$, where Δ is the Laplace-Beltrami operator on the manifold $\mathcal{M}$. Solutions to the heat equation are known in very few cases (Pennec, 2006), and even for the sphere the solution is only known as an infinite series of Legendre polynomials (Tulovsky & Papiez, 2001). Therefore using the SDE framework on manifolds will often require some numerical solutions to the relevant SDE/ODE. In contrast, our LMC-based formulation provides the flexibility to specify arbitrary target probability paths between the prior and data densities. On the sphere for example, we use closed form paths defined by vMF distributions. For sampling, solving an ODE is generally easier than solving an SDE as ODE solvers have higher asymptotic convergence rates. For example, Euler's method has order 1 for ODE and only 0.5 for SDE (Kloeden et al., 2012). Furthermore, ODEs have simple higher order solvers like Runga-Kutta methods (Dormand & Prince, 1980) with widely used open-source implementations.

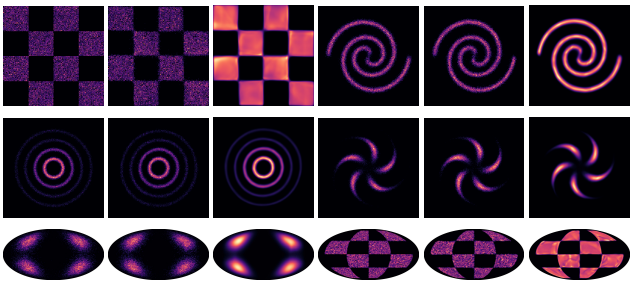

Figure 3. 2D toy densities. Each triplet shows (left to right): data samples, generate samples $x \sim q_1$, and learned model density q_1.

5. Experiments

We have tested the CNFM framework with the PPD for training CNFs on low and moderately high dimensional manifold data. In all experiments we generate the target path p according to Section 3.3 with input data samples $\{y_i\}_{i=1}^m \subset \mathcal{M}$. In all experiments we iterate over the dataset where the set $\{y_i\}_{i=1}^m \subset \mathcal{M}$, which is used for the approximation of equation 15, is simply the batch, that is $m = \text{batch size}$. Note that for better approximation of equation 15, we could take $m > \text{batch size}$ and evaluate the loss only at a subset of size batch size. Since the loss is still evaluated at only batch size of samples, i.e., the forward and backward costs are the same, and memory usage will not increase significantly. In general, we have found CNFM to facilitate faster training of CNFs with larger models, often producing state of the art sampling and density estimation.

5.1. Toy densities on $\mathbb{R}^2$ and $\mathcal{S}^2$

In the first experiment we worked with samples drawn from standard toy distributions on the 2D Euclidean plane and sphere. For the Euclidean data we used the target path p

Dataset	Earthquake	Flood	Fire	Volcano
Mixture vMF	0.59±0.01	1.09±0.01	−0.23±0.02	−0.31±0.07
Stereographic	0.43±0.04	0.99±0.04	−0.40±0.06	−0.64±0.20
Riemannian	0.19±0.04	0.90±0.03	−0.66±0.05	−0.97±0.15
Moser Flow	−0.09±0.02	0.62±0.04	−1.03±0.03	−2.02±0.42
CNFM	**−0.38**±0.01	**0.25**±0.02	**−1.40**±0.02	**−2.38**±0.16

Table 1. Negative log likelihood scores on the Earth and Climate Dataset (Mathieu & Nickel, 2020).

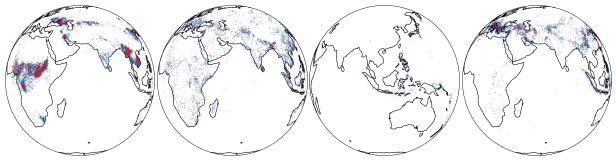

Figure 4. Earth and Climate dataset: generated samples from the trained CNFM in blue, test samples in red. See table 1 for quantitative results.

defined in equation 17 with $p_0 \sim \mathcal{N}(x|0, \boldsymbol{I})$, the standard normal distribution, and $\sigma_1 = 0.01$. For the spherical data we used the target path p as defined in equation 20 with $\kappa_1 = 5000$. We used MLP of 3 layers of 256 neurons for the $\mathbb{R}^2$ data, and 6 layers of 512 neurons for $\mathcal{S}^2$. We used PPD with $\ell = 1$. Figure 3 depicts the data samples y_i along side samples generated from the learned model, and the model densities. Note the high similarity between the learned and GT densities; for sphere visualizations we use Mollweide projection.

5.2. Earth and climate dataset

In this experiment we considered the Earth and Climate dataset curated in (Mathieu & Nickel, 2020). This dataset contains locations of earthquakes, floods, fires, and volcano eruptions on earth, represented as point locations on the 2D sphere, $\mathcal{S}^2 \subset \mathbb{R}^3$. The target path p is defined as in equation 20 with $\kappa_1 = 55$K (best out of $\kappa_1 \in \{5\text{K}, 55\text{K}, 500\text{K}\}$). We used the same architecture used in (Rozen et al., 2021), a MLP with 6 layers of 512 neurons, PPD order $\ell = 2$. Table 1 depicts the negative log likelihoods (NLLs) scores, where CNFM improves state of the art by a large margin, where the runner-up is Moser Flow (Rozen et al., 2021). Riemannian CNF and other baselines are taken from (Mathieu & Nickel, 2020). Figure 4 visualizes generated samples (blue) and test data samples (red).

5.3. Higher dimensional spheres

In this experiment we test the scaling of CNFM to higher dimensional manifold data. We construct a family of challenging probability distributions, denoted r_k, on $\mathcal{S}^{15}$ and

Figure 5. Left triplet shows the densities r_k for $k = 2, 3, 4$ on random cuts $\mathcal{S}^2 \subset \mathcal{S}^{15}$; right triplet visualizes the case $k = 3$ (on a different random cut) from Table 2 with CNFM model density in the middle, and S-FFJORD density on the right.

compare CNFM to several baselines. We start by defining r_k over $\mathcal{S}^{15} \subset \mathbb{R}^{16}$: Henceforth, denote $d = 15$, and consider an orthogonal set $v_1, \ldots, v_k$, where $1 \leq k \leq d+1$. Let $s(x) = \prod_{i=1}^{k} \text{sign}(x^T v_i)$. Define the probability density:

$$r_k(x) = \frac{2}{|\mathcal{S}^d|} \begin{cases} 1 & \text{if } s(x) = 1 \\ 0 & \text{if } s(x) = -1 \end{cases} \tag{23}$$

To see r_k is indeed a probability density, note that the transformation $x = (x_1, \ldots, x_{d+1}) \mapsto (-x_1, \ldots, x_{d+1})$ is a volume preserving transformation of $\mathcal{S}^d$ and maps the set $\Omega_+ = \mathcal{S}^d \cap \left\{x \in \mathbb{R}^{d+1} | s(x) = 1\right\}$ to $\Omega_- = \mathcal{S}^d \cap \left\{x \in \mathbb{R}^{d+1} | s(x) = -1\right\}$, and vise versa. This means that $\int_{\Omega_+} dV_x = \int_{\Omega_-} dV_x$ and since $\mathcal{S}^d = \Omega_+ \cup \Omega_-$ we have that $\int_{\Omega_+} dV_x = \left|\mathcal{S}^d\right|/2$. Generating samples from r_k can be done by randomizing a uniform sample x over $\mathcal{S}^d$, if $s(x) = 1$, keep x, otherwise take $(-x_1, x_2, \ldots, x_{d+1})$. Figure 5-left depicts several examples of this density by visualizing random $\mathcal{S}^2$ cuts in $\mathcal{S}^{15}$; as k increases the complexity of density increases. We created datasets for $k = 2, 3, 4$ with 45K train samples and 5K test samples. For baselines we use: vMF mixture models (vMF-MM) with 1K and 10K centers randomized from the training data, and scaling κ was chosen to be the optimal for the test set. This was done to compare to the best possible vMF-MM model. Furthermore, we compared to a version of manifold CNF (Lou et al., 2020; Mathieu & Nickel, 2020; Falorsi & Forré, 2020): We consider the stereographic projection of the sphere $\Psi : \mathbb{R}^d \to \mathcal{S}^d$, and used FFJORD (Grathwohl et al., 2018) code adapted to the spherical case, denoted as S-FFJORD. In this baseline, computing log probabilities over the sphere is done by correcting for the stereographic projection, $\log p(\Psi(u)) = \log p(u) - \frac{1}{2} \log \det(D_\Psi(u)^T D_\Psi(u))$, where $u \in \mathbb{R}^d$, $\log p(u)$ is the Euclidean log probability learned by FFJORD, $D_\Psi(u) \in \mathbb{R}^{(d+1)\times d}$ is the matrix of partials of Ψ. Table 2 reports the NLL scores of CNFM and the baselines across this dataset. Figure 5-right depicts an example of random $\mathcal{S}^2$ cut of $\mathcal{S}^{15}$ for the $k = 3$ case.

	2	3	4
vMF-MM	1.23	1.31	1.33
S-FFJORD	0.77	0.97	1.04
CNFM	**0.73**	**0.83**	**0.95**

Table 2. NLLs on $\mathcal{S}^{15}$.

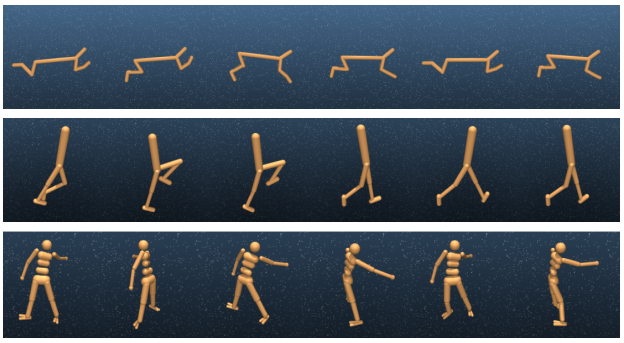

Figure 7. Uncurated samples computed with the trained CNFM on product manifolds representing the robot's state space: Cheetah (top), Walker (middle), and Humanoid (bottom).

Figure 8. Noise to data paths computed with the trained CNFM on product manifolds representing robot's state space: Cheetah (top), Walker (middle), and Humanoid (bottom).

The above results are reported using an MLP with 3 layers of 64 neurons for CNFM and an equivalent architecture of S-FFJORD, with both methods running for about 4K seconds. In Figure 6 we compared typical epoch running times for this and larger architecture types for CNFM and FFJORD training. Note that the time difference (in log scale) further increases for larger architectures.

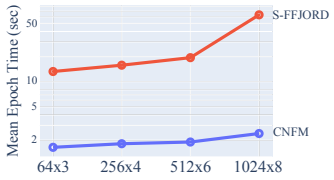

Figure 6. Timings.

5.4. Product of manifolds - Robotics

In the last experiment we worked with robotics data generated with the physics and reinforcement learning engine MuJoCo (Tassa et al., 2020). For each of the three robot types, Walker (2D), Cheetah (2D), and Humanoid (3D), we randomized 17.5K samples from 50 simulated trajectories consisting of 500 observation each. The state space for the 2D robots is modeled as the product manifold $\mathcal{M} = \mathbb{R}^3 \times (\mathcal{S}^1)^6$, where $\mathbb{R}^3$ represents position, and $\mathcal{S}^1$ represents 2D rotations of a single joint. The state space for the 3D robot is $\mathcal{M} = \mathbb{R}^3 \times (\mathcal{S}^1)^8 \times (\mathcal{S}^3)^6$, where $\mathcal{S}^3$ represents 3D rotations of a joint (via quaternions). We used the target path p on the product manifold as described in Section 3.3. For each robot type, Figure 7 depicts uncurated samples from the trained CNFM, and Figure 8 shows a path of noise to data, i.e., $\phi_t(x)$, $t \in [0, 1]$, where $x \sim p_0$. The generated samples are qualitatively similarly to data samples. More examples are in Appendix G.

6. Limitations and Future Work

Scaling CNFM to even higher dimensions (i.e., $d > 100$), e.g., for image data, requires some more work. We identify two main challenges: (i) using `logsumexp` and stochastic approximation for $\mathrm{div}(v)$ introduces a non-trivial approximation error (and bias) to the gradient estimation of the PPD; and (ii) the PPD loss has very different scales for different values of t, which entails conditioning. Using CNFM as-is on the MNIST dataset ($d = 784$) with a standard batch size of 128 results in samples shown in inset. Although generation quality does not match SOTA CNF models, the training process remains stable despite the high dimensional biased gradient estimation of the loss. We leave scaling CNFM to images to future work.

7. Conclusions

We have introduced CNFM, a framework for matching a target density path and the density path generated by a CNF. The CNFM is based on minimizing a novel Probability Path Divergence (PPD) that does not require sampling of model densities and therefore is easier to train and to apply to manifolds. The PPD is shown to upper bound standard divergences, and can work with a rather flexible family of target paths on manifolds. Empirically, CNFM was shown to facilitate CNF training, scaling for the first time to manifolds of moderate dimension, improving training time, and producing state of the art samplings and log likelihoods.

References

Ali, S. M. and Silvey, S. D. A general class of coefficients of divergence of one distribution from another. *Journal of the Royal Statistical Society: Series B (Methodological)*, 28(1):131–142, 1966.

Bogachev, V. I. *Measure theory*, volume 1. Springer Science & Business Media, 2007.

Bose, A. J., Smofsky, A., Liao, R., Panangaden, P., and Hamilton, W. L. Latent Variable Modelling with Hyperbolic Normalizing Flows. *arXiv:2002.06336 [cs, stat]*, February 2020. URL http://arxiv.org/abs/2002.06336.

Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. Neural ordinary differential equations. *arXiv preprint arXiv:1806.07366*, 2018.

Csiszár, I. Information-type measures of difference of probability distributions and indirect observation. *studia scientiarum Mathematicarum Hungarica*, 2:229–318, 1967.

Dormand, J. R. and Prince, P. J. A family of embedded runge-kutta formulae. *Journal of computational and applied mathematics*, 6(1):19–26, 1980.

Evans, L. C. Partial differential equations and monge-kantorovich mass transfer. *Current developments in mathematics*, 1997(1):65–126, 1997.

Falorsi, L. and Forré, P. Neural Ordinary Differential Equations on Manifolds. *arXiv:2006.06663 [cs, stat]*, June 2020. URL http://arxiv.org/abs/2006.06663.

Grathwohl, W., Chen, R. T. Q., Bettencourt, J., Sutskever, I., and Duvenaud, D. Ffjord: Free-form continuous dynamics for scalable reversible generative models, 2018.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *arXiv preprint arXiv:2006.11239*, 2020.

Kloeden, P. E., Platen, E., and Schurz, H. *Numerical solution of SDE through computer experiments*. Springer Science & Business Media, 2012.

Lou, A., Lim, D., Katsman, I., Huang, L., Jiang, Q., Lim, S.-N., and De Sa, C. Neural manifold ordinary differential equations. 2020.

Mardia, K. V. *Statistics of directional data*. Academic press, 2014.

Mardia, K. V., Hughes, G., Taylor, C. C., and Singh, H. A multivariate von mises distribution with applications to bioinformatics. *The Canadian Journal of Statistics / La Revue Canadienne de Statistique*, 36(1):99–109, 2008. ISSN 03195724. URL http://www.jstor.org/stable/20445295.

Mathieu, E. and Nickel, M. Riemannian continuous normalizing flows. *arXiv preprint arXiv:2006.10605*, 2020.

Oh, C., Adamczewski, K., and Park, M. Radial and directional posteriors for bayesian neural networks. *arXiv preprint arXiv:1902.02603*, 2019.

Pennec, X. Intrinsic statistics on riemannian manifolds: Basic tools for geometric measurements. *Journal of Mathematical Imaging and Vision*, 25(1):127, 2006.

Rezende, D. J. and Mohamed, S. Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pp. 1530–1538. JMLR.org, 2015.

Rezende, D. J., Papamakarios, G., Racaniere, S., Albergo, M., Kanwar, G., Shanahan, P., and Cranmer, K. Normalizing flows on tori and spheres. In *International Conference on Machine Learning*, pp. 8083–8092. PMLR, 2020.

Rozen, N., Grover, A., Nickel, M., and Lipman, Y. Moser flow: Divergence-based generative modeling on manifolds. *arXiv preprint arXiv:2108.08052*, 2021.

Ruiz-Antolín, D. and Segura, J. A new type of sharp bounds for ratios of modified bessel functions. *Journal of Mathematical Analysis and Applications*, 443(2):1232–1246, 2016.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pp. 2256–2265. PMLR, 2015.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. *arXiv preprint arXiv:1907.05600*, 2019.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Tassa, Y., Tunyasuvunakool, S., Muldal, A., Doron, Y., Liu, S., Bohez, S., Merel, J., Erez, T., Lillicrap, T., and Heess, N. dm_control: Software and tasks for continuous control, 2020.

Tulovsky, V. and Papiez, L. Formula for the fundamental solution of the heat equation on the sphere. *Applied mathematics letters*, 14(7):881–884, 2001.

Villani, C. *Optimal transport: old and new*, volume 338. Springer, 2009.

A. Proof of Theorem 1

Since $v \in \mathfrak{X}(\mathcal{M})$ it is locally Lipschitz. Since v is bounded it satisfies in particular

$$\int_0^1 \int_{\mathcal{M}} |v(t,x)|\, p_t(x)\, dV_x dt < +\infty.$$

Therefore, according to the Mass Conservation Formula Theorem (see e.g., (Villani, 2009)) equation 4 holds iff

$$\partial_t p_t + \operatorname{div}(p_t v) = 0, \tag{24}$$

where div denotes the divergence operator over the manifold $\mathcal{M}$. We assumed $p_t > 0$ and therefore dividing both sides with p_t leads to

$$\frac{\partial_t p_t}{p_t} + \frac{\langle \nabla_x p_t, v\rangle + p_t \operatorname{div}(v)}{p_t} = 0.$$

where we used that $\operatorname{div}(fv) = \langle \nabla_x f, v\rangle + f\operatorname{div}(v)$ for $f \in \mathfrak{P}(\mathcal{M})$ and $v \in \mathfrak{X}(\mathcal{M})$. Finally noting that $\partial_t \log p_t = \frac{\partial_t p_t}{p_t}$, and $\nabla_x \log p_t = \frac{\nabla_x p_t}{p_t}$ we get that equation 5 is equivalent to equation 24. □

B. Proof of Lemma 1

Given a time dependent vector field $v \in \mathfrak{X}(\mathcal{M})$, a diffeomorphism two parameter family $\Phi_{t,t_0} : \mathcal{M} \to \mathcal{M}$ can be defined via the following Ordinary Differential Equation (ODE):

$$\begin{cases} \frac{d}{dt}\Phi_{t,t_0}(x) = v(t, \Phi_{t,t_0}(x)) \\ \Phi_{t_0,t_0}(x) = x \end{cases} \tag{25}$$

The CNF diffeomorphism is defined by $\phi_t = \Phi_{t,0}$. Now, consider a smooth function $u(t,x)$, $u : [0,1] \times \mathcal{M} \to \mathbb{R}$, then

$$\partial_t|_{t=t_0} \left[u(t, \Phi_{t,t_0}(x))\right] = \partial_t u(t_0, x) + \langle \nabla_x u(t_0, x), v(t_0, x)\rangle\,. \tag{26}$$

From Theorem 1 we have that

$$\partial_t \log q_t + \langle \nabla_x \log q_t, v\rangle + \operatorname{div}(v) = 0$$

for all $t \in [0,1]$ and $x \in \mathcal{M}$. Subtracting that in our loss we get

$$\mathrm{d}_\ell(p \,\|\, q) = \mathbb{E}_{t,x\sim p_t} \left| \partial_t \log \frac{p_t}{q_t} + \left\langle \nabla_x \log \frac{p_t}{q_t}, v \right\rangle \right|^\ell \tag{27}$$

Now using equation 26 with $u(t,x) = \log \frac{p_t(x)}{q_t(x)}$, we get

$$\partial_t|_{t=t_0} \log \frac{p_t(\Phi_{t,t_0}(x))}{q_t(\Phi_{t,t_0}(x))} =$$
$$\partial_t \log \frac{p_{t_0}(x)}{q_{t_0}(x)} + \left\langle \nabla_x \log \frac{p_{t_0}(x)}{q_{t_0}(x)}, v(t_0,x) \right\rangle$$

Plugging this in equation 27 with $t = s$ and $t_0 = t$ we get:

$$\begin{aligned}
\mathrm{d}_\ell(p \,\|\, q) &= \mathbb{E}_{t,x\sim p_t} \left| \partial_s|_{s=t} \log \frac{p_s(\Phi_{s,t}(x))}{q_s(\Phi_{s,t}(x))} \right|^\ell \\
&= \mathbb{E}_{t,x\sim q_t} \frac{p_t(x)}{q_t(x)} \left| \partial_s|_{s=t} \log \frac{p_s(\Phi_{s,t}(x))}{q_s(\Phi_{s,t}(x))} \right|^\ell \\
&= \mathbb{E}_{t,x\sim p_0} \frac{p_t(\phi_t(x))}{q_t(\phi_t(x))} \left| \partial_s|_{s=t} \log \frac{p_s(\Phi_{s,t}(\phi_t(x)))}{q_s(\Phi_{s,t}(\phi_t(x)))} \right|^\ell \\
&= \mathbb{E}_{t,x\sim p_0} \frac{p_t(\phi_t(x))}{q_t(\phi_t(x))} \left| \partial_s|_{s=t} \log \frac{p_s(\phi_s(x))}{q_s(\phi_s(x))} \right|^\ell \\
&= \mathbb{E}_{x\sim p_0} \int_0^1 \frac{p_t(\phi_t(x))}{q_t(\phi_t(x))} \left| \partial_t \log \frac{p_t(\phi_t(x))}{q_t(\phi_t(x))} \right|^\ell dt
\end{aligned}$$

where in the third equality we used the fact that $\phi_t(x) \sim q_t$ if $x \sim p_0$; in the fourth equality we used the fact that $\Phi_{s,t}(\phi_t(x)) = \Phi_{s,t}(\Phi_{t,0}(x)) = \Phi_{s,0}(x) = \phi_s(x)$. □

C. Additional details for the proof of Theorem 2

We add here details of the proof of Theorem 2 missing from the main paper.

First, we prove that for any $t > 0$, $\ell(1 - t^{\frac{1}{\ell}}) \nearrow -\log(t)$, that is monotonically increasing and converging to $-\log(t)$ as $\ell \to \infty$. Fix $t > 0$, and define the function

$$f(s) = \frac{(1 - t^s)}{s}$$

where $s \in (0, 1)$. Now, using L'Hôpital's rule:

$$\lim_{s \downarrow 0} f(s) = \lim_{s \downarrow 0} \frac{-\log(t)t^s}{1} = -\log(t)$$

Therefore in particular $\lim_{\ell \to \infty} \ell(1 - t^{\frac{1}{\ell}}) = -\log(t)$. Monotonicity follows from the fact that for all $s \in (0, 1)$ and $t > 0$

$$f'(s) = \frac{-\log(t)st^s + t^s - 1}{s^2} = \frac{t^s}{s}\left(\frac{1 - t^{-s}}{s} - \log(t)\right) \leq 0$$

The inequality can be justified by first noting that $t^s/s > 0$. Second, let $0 < t = \exp(r)$ we get that

$$\frac{1 - t^{-s}}{s} - \log(t) \leq 0$$

which is true iff

$$\frac{1 - \exp(-rs)}{s} - r \leq 0$$

which is true iff

$$1 - rs \leq \exp(-rs)$$

which is true iff for all $u \in \mathbb{R}$

$$1 - u \leq \exp(-u)$$

which is true since $1 - u$ is tangent to $\exp(-u)$ at $u = 0$, and $\exp(u)$ is convex. Since $f'(s)$ is monotonically decreasing in s, $\ell(1 - t^{\frac{1}{\ell}})$ is increasing as $\ell \to \infty$.

We are now ready to justify equation 13. First let

$$f_\ell(x) = \ell\left(1 - \left[\frac{p_T(x)}{q_T(x)}\right]^{\frac{1}{\ell}}\right)$$

We showed that $f_\ell(x) \nearrow f(x) = -\log \frac{p_T(x)}{q_T(x)}$. Furthermore, f_ℓ are all integrable since

$$\begin{aligned}
&\int_{\mathcal{M}} \ell \left|1 - \left[\frac{p_T(x)}{q_T(x)}\right]^{\frac{1}{\ell}}\right| q_T(x) dV_x \\
&\leq \ell \int_{\mathcal{M}} q_T(x) dV_x + \ell \int_{\mathcal{M}} p_T(x)^{\frac{1}{\ell}} q_T^{1-\frac{1}{\ell}} dV_x \\
&\leq \ell + \ell \left[\int_{\mathcal{M}} p_T(x) dV_x\right]^{\frac{1}{\ell}} \left[\int_{\mathcal{M}} q_T(x) dV_x\right]^{1-\frac{1}{\ell}} \\
&= 2\ell
\end{aligned}$$

Where in the first inequality we used the triangle inequality, and in the second inequality we used Holder inequality with $\frac{1}{\ell} + \frac{\ell-1}{\ell} = 1$.

We assume that

$$D_f(p_T \,\|\, q_T) = \int_{\mathcal{M}} f(x) q_T(x) dV_x < \infty.$$

Since $f_\ell(x) \leq f(x)$ and both f_ℓ, f are integrable we have that

$$\int_{\mathcal{M}} f_\ell(x) q_T(x) dV_x \leq \int_{\mathcal{M}} f(x) q_T(x) dV_x < \infty$$

for all ℓ. Therefore, the Monotone Convergence Theorem (see Theorem 2.8.2 in (Bogachev, 2007)) implies that

$$\lim_{\ell \to \infty} \int_{\mathcal{M}} f_\ell(x) q_T(x) dV_x = \int_{\mathcal{M}} f(x) q_T(x) dV_x$$

Namely,

$$\lim_{\ell \to \infty} \mathbb{E}_{x \sim q_T} \ell \left(1 - \left[\frac{p_T(x)}{q_T(x)} \right]^{\frac{1}{\ell}} \right) = -\mathbb{E}_{x \sim q_T} \log \frac{p_T(x)}{q_T(x)}$$

D. Numerically stable derivative of the normalizing constant of the vMF

The log of the normalizing constant of the vMF has the form

$$\log C_p(\kappa) = \left(\frac{p}{2} - 1 \right) \log \kappa - \frac{p}{2} \log(2\pi) - \left[\kappa + \log \texttt{ive}(\frac{p}{2} - 1, \kappa) \right]$$

where $\texttt{ive}(\nu, \kappa) = \texttt{iv}(\nu, \kappa) \exp(-\kappa)$. Now, the $\log \texttt{ive}(\frac{p}{2} - 1, \kappa)$ is stable but its derivative is not. Therefore we will define a new function and its derivative: $\texttt{logive}(\nu, \kappa)$. Its forward will be defined by;

$$\texttt{logive}(\nu, \kappa) = \log \left(\texttt{ive}(\nu, \kappa) \right),$$

and for its derivative we first note:

$$\begin{aligned} \partial_\kappa \log \texttt{ive}(\nu, \kappa) &= \frac{\partial_\kappa \texttt{ive}(\nu, \kappa)}{\texttt{ive}(\nu, \kappa)} = \frac{\texttt{ive}(\nu - 1, \kappa) - \texttt{ive}(\nu, \kappa) \left[\frac{\nu + \kappa}{\kappa} \right]}{\texttt{ive}(\nu, \kappa)} \\ &= \frac{\texttt{ive}(\nu - 1, \kappa)}{\texttt{ive}(\nu, \kappa)} - \left[\frac{\nu + \kappa}{\kappa} \right] = \frac{\texttt{ive}(\nu - 1, \kappa)}{\texttt{ive}(\nu, \kappa)} - \frac{\nu}{\kappa} - 1 \end{aligned}$$

For high dimensions the $\texttt{ive}$ ratio is numerically unstable and several approximations have been suggested. In particular (Ruiz-Antolín & Segura, 2016) suggest the following lower and upper bounds:

$$\frac{\nu - \frac{1}{2} + \sqrt{(\nu + \frac{1}{2})^2 + \kappa^2}}{\kappa} > \frac{\texttt{ive}(\nu - 1, \kappa)}{\texttt{ive}(\nu, \kappa)} > \frac{\nu - 1 + \sqrt{(\nu + 1)^2 + \kappa^2}}{\kappa}$$

Similar to (Oh et al., 2019) we take the average of the higher and lower bound (see (Oh et al., 2019) for empirically demonstrating the quality of this approximation):

$$\partial_\kappa \log \texttt{ive}(\nu, \kappa) = \frac{\texttt{ive}(\nu - 1, \kappa)}{\texttt{ive}(\nu, \kappa)} - \frac{\nu}{\kappa} - 1 \approx \frac{-1.5 + \sqrt{(\nu + 1)^2 + \kappa^2} + \sqrt{(\nu + \frac{1}{1})^2 + \kappa^2}}{2\kappa} - 1$$

and this is defined as the derivative of $\texttt{logive}$.

E. Vector field representation

We describe how we represent the parametric part of our system, namely the vector field $v_\theta \in \mathfrak{P}(\mathcal{M})$, for the different manifold types we consider: Euclidean space, spheres and product manifolds. In the *Euclidean* case we use an MLP $v_\theta : \mathbb{R}^{d+1} \to \mathbb{R}^d$, where d is the dimension of $\mathcal{M}$. We use standard Euclidean inner product, gradient ∇, and divergence $\mathrm{div} = \nabla\cdot$ for computing the PPD (equation 7), where the path p defined in equation 17. In the *sphere* case $\mathcal{S}^d \subset \mathbb{R}^{d+1}$, similarly to (Rozen et al., 2021) we define v to be constant in the normal direction to the sphere and produce tangent vectors via the tangent projection operator

$$v_\theta(t, x) = \left(\boldsymbol{I} - \frac{xx^T}{\|x\|^2}\right) w_\theta\left(t, \frac{x}{\|x\|}\right), \tag{28}$$

where $w_\theta : \mathbb{R}^{d+2} \to \mathbb{R}^{d+1}$ is an MLP. The inner product on the sphere is the induced Euclidean one, i.e., for $v, u \in T_x\mathcal{S}^d$ we have $\langle u, v\rangle = u^T v$. For v_θ defined in equation 28, the Riemannian gradient and divergence coincide with the Euclidean gradient ∇ and divergence $\nabla\cdot$. p is defined as in equation 20. For notational simplicity we explain the *product manifold* implementation for two manifolds $\mathcal{M} = \mathbb{R}^{d_1} \times \mathcal{S}^{d_2}$, where the extension to product of N manifolds is similar. The tangent vector field is a function of the form $v_\theta : \mathbb{R}^{d_1+1} \times \mathbb{R}^{d_2+2} \to \mathbb{R}^{d_1} \times \mathbb{R}^{d_2+1}$. For $(t_1, x_1, t_2, x_2) \in \mathbb{R}^{d_1+1} \times \mathbb{R}^{d_2+2}$ we let

$$v_\theta(t_1, x_1, t_2, x_2) = \begin{bmatrix} v_1\left(t_1, x_1, t_2, \frac{x_2}{\|x_2\|}\right) \\ \left(\boldsymbol{I} - \frac{x_2 x_2^T}{\|x_2\|^2}\right) v_2\left(t_1, x_1, t_2, \frac{x_2}{\|x_2\|}\right) \end{bmatrix}$$

where v_1, v_2 are MLPs. The inner product $(v_1, v_2), (u_1, u_2) \in T_x\mathcal{M}$ is defined by $\langle v_1, u_1\rangle + \langle v_2, u_2\rangle$; the gradient as $\nabla = (\nabla_1, \nabla_2)^T$, where ∇_1 is the Euclidean gradient w.r.t. x_1, and ∇_2 is the Euclidean gradient w.r.t. x_2; the divergence $\mathrm{div}(v) = \nabla_1 \cdot v_1 + \nabla_2 \cdot v_2$. Lastly, p is defined as in equation 15 with kernel equation 21.

F. Experimental Details

F.1. Toy densities on $\mathbb{R}^2$ and $\mathcal{S}^2$

For the $\mathbb{R}^2$ datasets we used a 3 layer MLP with hidden dimension 256. We trained with Adam optimizer with learning rate $1e-4$, batch size 1000, $\sigma_1 = 0.01$ and $\ell = 1$. The searched parameters across learning rates are $\{1e-3, 5e-4, 1e-4\}$ and $\sigma_1 \in \{0.005, 0.01, 0.05\}$. For the $\mathcal{S}^2$ datasets we used a 6 layer MLP with hidden dimension 512. We trained with Adam optimizer with learning rate $1e-4$, batch size 1000, $\kappa = 5000$ and $\ell = 1$.

F.2. Earth and climate datasets

For the earth and climate datasets we used a 6 layer MLP with hidden dimension 512. We trained with Adam optimizer with learning rate $1e-4$, batch size 1000, $\kappa = 55K$ and $\ell = 2$. The searched parameters across κ are $\{5K, 55K, 500k\}$.

F.3. Higher dimensional spheres

We ran experiments on $\mathcal{S}^{15}$ for different $k = 2, 3, 4$ values. The architecture used was a 3 layer MLP with hidden dimension 64, Adam optimizer with learning rate $1e-3$, batch size 7000, $\kappa = 5K$ and $\ell = 2$. We searched over learning rates $\{1e-3, 1e-4, 1e-5\}$.

The S-FFJORD baseline is as described in the paper. We used the architecture used for the 2D toy experiments in the FFJORD paper, as published in the official FFJORD code repository. We run both CNFM and S-FFJORD with approximate divergence computation using the Hutchinson estimator.

F.4. Product of manifolds - Robotics

For the robotics datasets we used a 6 layer MLP with hidden dimension 512. We trained with Adam optimizer with learning rate $1e-4$, batch size 1000, $\kappa = 55K$ and $\ell = 1$. The searched parameters across κ are $\{5K, 55K\}$.

G. Extra Experimental Results

We provide more uncurated samples and interpolations of Cheetah, Walker and Humanoid poses in Figure 9 and Figure 10.

Figure 9. Uncurated samples computed with the trained CNF on product manifolds representing robot's state space: Cheetah (top), Walker (middle), and Humanoid (bottom).

Figure 10. Noise to data paths computed with the trained CNF on product manifolds representing robot's state space:Cheetah (top), Walker (middle), and Humanoid (bottom).