f ff

Principled Interpolation in Normalizing Flows

Samuel G. Fadel (✉) [*1,2[0000−0002−4459−4336]],
Sebastian Mair[*2[0000−0003−2949−8781]],
Ricardo da S. Torres[3[0000−0001−9772−263X]], and Ulf Brefeld[2]

[1] University of Campinas, Brazil
samuel.fadel@ic.unicamp.br
[2] Leuphana University, Germany
{mair,brefeld}@leuphana.de
[3] Norwegian University of Science and Technology, Norway
ricardo.torres@ntnu.no

Abstract. Generative models based on normalizing flows are very successful in modeling complex data distributions using simpler ones. However, straightforward linear interpolations show unexpected side effects, as interpolation paths lie outside the area where samples are observed. This is caused by the standard choice of Gaussian base distributions and can be seen in the norms of the interpolated samples as they are outside the data manifold. This observation suggests that changing the way of interpolating should generally result in better interpolations, but it is not clear how to do that in an unambiguous way. In this paper, we solve this issue by enforcing a specific manifold and, hence, change the base distribution, to allow for a principled way of interpolation. Specifically, we use the Dirichlet and von Mises-Fisher base distributions on the probability simplex and the hypersphere, respectively. Our experimental results show superior performance in terms of bits per dimension, Frchet Inception Distance (FID), and Kernel Inception Distance (KID) scores for interpolation, while maintaining the generative performance.

Keywords: generative modeling · density estimation · normalizing flows

1 Introduction

Learning high-dimensional densities is a common task in unsupervised learning. Normalizing flows [32, 31, 28, 9, 26] provide a framework for transforming complex distributions into simple ones: a chain of L parametrized bijective functions $\mathbf{f} = \mathbf{f}_1 \circ \mathbf{f}_2 \circ \cdots \circ \mathbf{f}_L$ converts data into another representation that follows a given base distribution. The likelihood of the data can then be expressed as the likelihood of the base distribution and the determinants of the Jacobians of the transformations $\mathbf{f}_i$. In contrast to generative adversarial networks (GANs) [12], the likelihood of the data can be directly optimized, leading to a straightforward training procedure. Moreover, unlike other approaches, such as variational autoencoders (VAEs) [17, 18], there is no reconstruction error since all functions $\mathbf{f}_i$ within this chain are bijections.

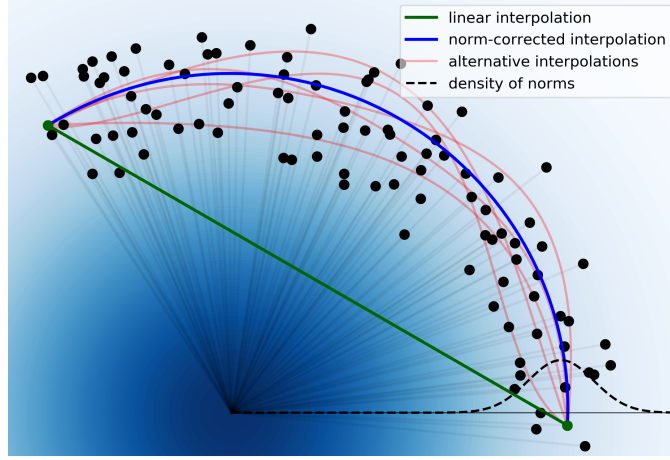
[*] Equal contribution.

Fig. 1. Illustration of different interpolation paths of points from a high-dimensional Gaussian. The figure also shows that, in high dimensions, points are not concentrated at the origin.

In flow-based generative models, data are generated by drawing samples from a base distribution, where the latter is usually given by a simple distribution, such as a standard Gaussian [23]. The Gaussian samples are then mapped to real data using the chain **f**. A prevalent operation is to linearly interpolate samples and consider the interpolation path in data space. In generative modeling, interpolations are frequently used to evaluate the quality of the learned model and to demonstrate that the model generalizes beyond what was seen in the training data [25].

The consequences for interpolation, however, are not immediately apparent for Gaussian base distributions. Figure 1 shows a linear interpolation (*lerp*) of high-dimensional samples from a Gaussian. The squared Euclidean norms of the samples follow a χ_d^2-distribution as indicated by the dashed black line. Data points have an expected squared Euclidean norm of length d, where d is the dimensionality. This implies that there is almost no point around the origin. As seen in the figure, the norms of a linear interpolation path (green line) of two samples drop significantly and lie in a low-density area w.r.t. the distribution of the norms (dashed black line) [33].

Instead of a linear interpolation (green line), an interpolation that preserves the norm distribution of interpolants is clearly preferable (blue and red lines): the blue and red interpolation paths stay in the data manifold and do not enter low-density areas. The observation suggests that interpolated samples with norms in a specific range should generally result in better interpolations. This can be achieved, i.e., by shrinking the variance of the density or norms (dashed lines), which yields a subspace or manifold with a fixed norm.
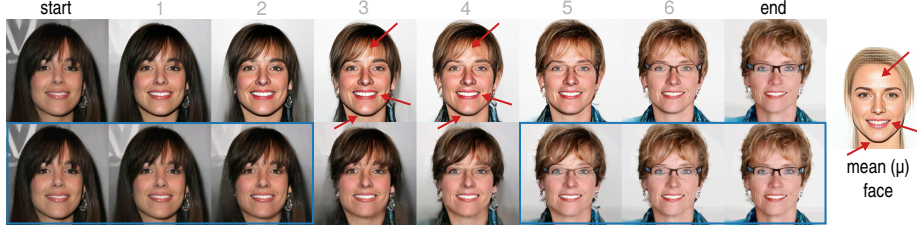
Fig. 2. Interpolation of samples from CelebA. *Top*: a linear interpolation path. The central images resemble features of the mean face as annotated in red. *Bottom*: an alternative interpolation path using a norm-correction. Note that the first and last three images are almost identical as annotated in blue. *Right*: decoded expectation of base distribution, i.e., the mean face.

In this paper, we propose a framework that respects the norm of the samples and allows for a principled interpolation, addressing the issues mentioned above. We study base distributions on supports that have a fixed norm. Specifically, we consider unit p-norm spheres for $p \in \{1, 2\}$, leading to the Dirichlet ($p = 1$) and the von Mises-Fisher ($p = 2$) distributions, respectively. The conceptual change naturally implies technical difficulties that arise with restricting the support of the base distribution to the simplex or the unit hypersphere. We thus need to identify appropriate bijective transformations into unit p-norm spheres.

The next sections are organized as follows. In Section 2, we propose a simple heuristic to the problem and discuss its problems before we introduce normalizing flows in Section 3. Section 4 contains the main contribution, a framework for normalizing flows onto unit p-norm spheres. Empirical results are presented in Section 5, and related work is discussed in Section 6. Section 7 provides our conclusions.

2   An Intuitive Solution

The blue path in Figure 1 is obtained by a norm correction of the linear interpolation via also interpolating the norms. Mathematically, that is

$$\gamma(\lambda) = \underbrace{((1 - \lambda)\mathbf{z}_a + \lambda\mathbf{z}_b)}_{\text{linear interpolation}} \cdot \underbrace{\frac{(1 - \lambda)\|\mathbf{z}_a\|_2 + \lambda\|\mathbf{z}_b\|_2}{\|(1 - \lambda)\mathbf{z}_a + \lambda\mathbf{z}_b\|_2}}_{\text{norm correction}}, \qquad (1)$$

for endpoints $\mathbf{z}_a, \mathbf{z}_b$ and $\lambda \in [0, 1]$. We refer to this approach as norm-corrected linear interpolation (*nclerp*). However, the depicted red lines also stay within the manifold, hence it remains unclear how a unique interpolation path can be obtained.

Figure 2 depicts two interpolation paths for faces taken from CelebA [14] created using Glow [19], a state-of-the-art flow-based model that uses a standard Gaussian as base distribution. The leftmost and rightmost faces of the paths are
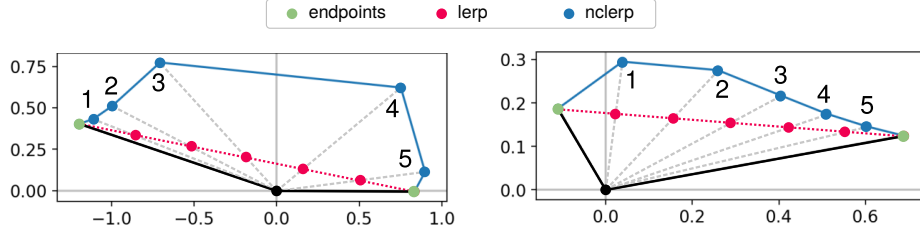
Fig. 3. Two examples showing the issues caused by a norm-corrected linear interpolation (*nclerp*).

real data, while the other ones are computed interpolants. The face on the right depicts the so-called *mean face*, which is given by the mean of the Gaussian base distribution and is trivially computed by decoding the origin of the space. The top row shows a linear interpolation similar to the green line in Figure 1. The interpolation path is close to the origin, and the interpolants consequentially resemble features of the mean face, such as the nose, mouth, chin, and forehead shine, which neither of the women have. We highlighted those features in red in Figure 2.

By contrast, the bottom row of Figure 2 shows the norm-corrected interpolation sequence (as the blue line in Figure 1): the background transition is smooth and not affected by the white of the mean face, and also subtleties like the shadow of the chin in the left face smoothly disappears in the transition. The norm correction clearly leads to a better transition from one image to the other. However, the simple heuristic in Equation (1) causes another problem: the path after norm correction is no longer equally spaced when λ values are equally spaced in $[0, 1]$. Implications of this can be seen in blue in the bottom row of Figure 2, where the first and last three faces are almost identical. We provide additional examples in the supplementary material.

In Figure 3, we illustrate two examples, comparing a linear interpolation (*lerp*) and a norm-corrected linear interpolation (*nclerp*) between points from a high-dimensional Gaussian (green points). For equally-spaced λ values in $[0, 1]$, a linear interpolation yields an equally-spaced interpolation path (red line). Evidently, the norm-corrected interpolation (blue line) keeps the norms of interpolants within the range observed in data.

However, the interpolants are no longer evenly spaced along the interpolation path. Hence, control over the interpolation mixing is lost. This problem is more pronounced on the left example, where points 1 through 3 are closer to the start point, while points 4 and 5 are closer to the endpoint. Consequently, evaluations such as Fréchet Inception Distance (FID) scores [13] will be affected. Such scores are computed by comparing two sets of samples, in this case, the real data and interpolated data. As those points will be clearly more similar to the endpoints, which are samples from the data set itself, the scores are in favor of the norm-corrected interpolation.
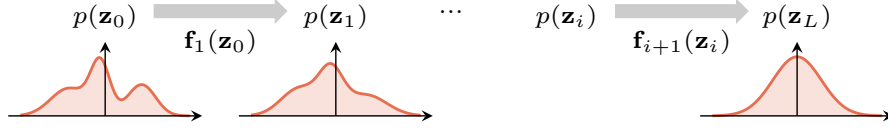
Fig. 4. An example of a one-dimensional normalizing flow.

3   Normalizing Flows

Let $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\} \subset \mathbb{R}^d$ be instances drawn from an unknown distribution $p_x(\mathbf{x})$. The goal is to learn an accurate model of $p_x(\mathbf{x})$. Let $\mathbf{f}^{(\theta)} : \mathbb{R}^d \to \mathbb{R}^d$ be a bijective function parametrized by θ. Introducing $\mathbf{z} = \mathbf{f}^{(\theta)}(\mathbf{x})$ and using the change of variable theorem allows us to express the unknown $p_x(\mathbf{x})$ by a (simpler) distribution $p_z(\mathbf{z})$, defined on $\mathbf{z} \in \mathbb{R}^d$, given by

$$p_x(\mathbf{x}) = p_z\left(\mathbf{f}^{(\theta)}(\mathbf{x})\right)\left|\det J_{\mathbf{f}}^{(\theta)}(\mathbf{x})\right|,$$

where $J_{\mathbf{f}}^{(\theta)}(\mathbf{x})$ is the Jacobian matrix of the bijective transformation $\mathbf{f}^{(\theta)}$. We denote $p_z(\mathbf{z})$ as the base distribution and drop the subscript of the distribution p whenever it is clear from the context.

Representing $\mathbf{f}^{(\theta)}$ as a chain of L parametrized bijective functions, i.e., $\mathbf{f}^{(\theta)} = \mathbf{f}_L^{(\theta)} \circ \mathbf{f}_{L-1}^{(\theta)} \circ \cdots \circ \mathbf{f}_1^{(\theta)}$, creates a *normalizing flow* that maps observations $\mathbf{x}$ into representations $\mathbf{z}$ that are governed by the base distribution $p(\mathbf{z})$. Let $\mathbf{z}_0 = \mathbf{x}$ be the input data point and $\mathbf{z}_L = \mathbf{z}$ be the corresponding output of the chain, where every intermediate variable is given by $\mathbf{z}_i = \mathbf{f}_i(\mathbf{z}_{i-1})$, where $i = 1, \dots, L$. A one-dimensional example is depicted in Figure 4. The data log-likelihood can then be expressed as the log-likelihood of the base distribution and the log-determinant of the Jacobians of each transformation as

$$\log p(\mathbf{x}) = \log p(\mathbf{z}) + \sum_{i=1}^{L} \log\left|\det J_{\mathbf{f}_i^{(\theta)}}(\mathbf{z}_{i-1})\right|.$$

Flow-based generative models can be categorized by how the Jacobian structure of each transformation $\mathbf{f}_i$ is designed since computing its determinant is crucial to its computational efficiency. The Jacobians either have a lower triangular structure, such as autoregressive flows [20], or a structured sparsity, such as coupling layers in RealNVP [10] and Glow [19]. Transformations with free-form Jacobians allow much higher expressibility by replacing the computation of the determinant with another estimator for the log-density [6]. For more information regarding flow-based generative models, we refer the reader to [23].

In the remainder, we simplify the notation by dropping the superscript θ from $\mathbf{f}$. We also note that a normalizing flow defines a generative process. To create a new sample $\mathbf{x}$, we first sample $\mathbf{z}$ from the base distribution $p(\mathbf{z})$ and then transform $\mathbf{z}$ into $\mathbf{x}$ using the inverse chain of transformations $\mathbf{f}^{-1}$.

4   Base Distributions on p-Norm Spheres

Motivated by earlier observations illustrated in Figures 1 and 2, we intend to reduce ambiguity by shrinking the variance of the norms of data. We achieve this by considering base distributions on restricted subspaces. More specifically, we focus on unit p-norm spheres defined by

$$\mathbb{S}_p^d = \left\{ \mathbf{z} \in \mathbb{R}^{d+1} \;\middle|\; \|\mathbf{z}\|_p^p = \sum_{j=1}^{d+1} |z_j|^p = 1 \right\}. \tag{2}$$

We distinguish two choices of p and discuss the challenges and desirable properties that ensue from their use. We consider $p \in \{1, 2\}$ as those allow us to use well-known distributions, namely the Dirichlet distribution for $p = 1$ and the von Mises-Fisher distribution for $p = 2$.

4.1   The Case p=1

For $p = 1$, the Dirichlet distribution defined on the standard simplex Δ^d is a natural candidate. Its probability density function is given by

$$p(\mathbf{s}) = \frac{1}{Z(\alpha)} \prod_{k=1}^{d+1} s_k^{\alpha_k - 1},$$

$$\text{with} \quad Z(\alpha) = \frac{\prod_{k=1}^{d+1} \Gamma(\alpha_k)}{\Gamma\left(\sum_{k=1}^{d+1} \alpha_k\right)},$$

where Γ is the gamma function and $\alpha_k > 0$ are the parameters. In order to make use of it, we also need to impose a non-negativity constraint in addition to Equation (2).

Let $\mathbf{z} \in \mathbb{R}^d$ be an unconstrained variable. The function $\phi : \mathbb{R}^d \to (0, 1)^d$ transforms $\mathbf{z}$ into a representation $\mathbf{s}$ by first transforming each dimension z_k into intermediate values v_k with

$$v_k = \sigma\left(z_k - \log(d + 1 - k)\right)$$

which are used to write $\mathbf{s}$ as

$$s_k = \left(1 - \sum_{l=1}^{k-1} s_l\right) \cdot v_k,$$

where $\sigma(\cdot)$ denotes the sigmoid function. We note a few details of this transformation. First, a property of ϕ is that $0 < \sum_{k=1}^d s_k < 1$. Therefore, a point in Δ^d can be obtained with an implicit additional coordinate $s_{d+1} = 1 - \sum_{k=1}^d s_k$. Second, the difference in dimensionality does not pose a problem for computing its Jacobian as ϕ establishes a bijection within $\mathbb{R}^d$ while the mapping to Δ^d is

given implicitly. Third, ϕ maps $\mathbf{z} = \mathbf{0}$ to the center of the simplex $\mathbf{s} = (d+1)^{-1}\mathbf{1}$. Fourth, since $\mathbf{s}$ consists of solely positive numbers which sum up to one, numerical problems may arise for high-dimensional settings. We elaborate on this issue in Section 5.

The Jacobian J_ϕ has a lower triangular structure and solely consists of non-negative entries. Hence, the log-determinant of this transformation can be efficiently computed in $\mathcal{O}(d)$ time via

$$
\log |\det J_\phi| = \sum_{k=1}^{d} \log \left(v_k \left(1 - v_k\right)\right) \\
+ \log \left(1 - \sum_{l=1}^{k-1} s_l\right).
$$

The inverse transformation $\phi^{-1} : \mathbb{R}^d \to \mathbb{R}^d$ is given by

$$
z_k = \sigma^{-1} \left(\frac{s_k}{1 - \sum_{l=1}^{k-1} s_l}\right) + \log(d + 1 - k).
$$

The interpolation of two points $\mathbf{a}, \mathbf{b} \in \Delta^d$ within the unit simplex is straightforward. A linear interpolation $(1 - \lambda)\mathbf{a} + \lambda\mathbf{b}$ using $\lambda \in [0, 1]$ is guaranteed to stay within the simplex by definition.

4.2   The Case p=2

For $p = 2$, data points lie on the surface of a d-dimensional hypersphere. The von Mises-Fisher (vMF) distribution, defined on $\mathbb{S}_2^d$, is frequently used in directional statistics. It is parameterized by a *mean direction* $\boldsymbol{\mu} \in \mathbb{S}_2^d$ and a *concentration* $\kappa \geq 0$, with a probability density function given by

$$
p(\mathbf{s}) = C_{d+1}(\kappa) \exp(\kappa \boldsymbol{\mu}^\top \mathbf{s}),
$$
$$
\text{with} \quad C_\nu(\kappa) = \frac{\kappa^{\nu/2-1}}{(2\pi)^{\nu/2} I_{\nu/2-1}(\kappa)},
$$

where I_w denotes the modified Bessel function of the first kind at order w.

Again, let $\mathbf{z} \in \mathbb{R}^d$ be an unconstrained variable. We employ a stereographic projection, for both its invertibility and its Jacobian, whose log-determinant can be efficiently computed. The transformation $\psi : \mathbb{R}^d \to \mathbb{S}_2^d$ maps a point $\mathbf{z} \in \mathbb{R}^d$ to a point $\mathbf{s} \in \mathbb{S}_2^d \subset \mathbb{R}^{d+1}$ on the hypersphere via

$$
\psi(\mathbf{z}) = \mathbf{s} = \begin{bmatrix} \mathbf{z}\rho_{\mathbf{z}} \\ 1 - \rho_{\mathbf{z}} \end{bmatrix}, \quad \text{with} \quad \rho_{\mathbf{z}} = \frac{2}{1 + \|\mathbf{z}\|^2}.
$$

The transformation ψ, which has no additional parameters, ensures that its image is on the unit hypersphere, allowing the use of a vMF distribution to model $p(\mathbf{s})$. Two points in $\mathbb{S}_2^d$ are of special interest, namely the *south pole* and the *north pole*, where the last coordinate of $\mathbf{s}$ is either -1 or 1, respectively. By construction, the transformation is symmetric around zero and sends $\mathbf{z} = \mathbf{0}$ to the *south pole*, which we choose as the mean direction $\boldsymbol{\mu}$. Furthermore, it is bijective up to an open neighborhood around the *north pole*, as $\rho_{\mathbf{z}} \to 0$ whenever $\|\mathbf{z}\|^2 \to \infty$. For this reason, we avoid choosing a uniform distribution on the hypersphere, which is obtained for $\kappa = 0$. Figure 5 shows an example.
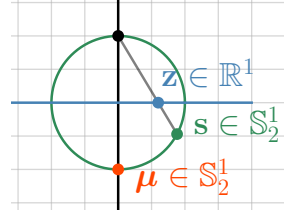

Fig. 5. A stereographic projection mapping $\mathbf{z} \in \mathbb{R}^1$ to $\mathbf{s} \in \mathbb{S}_2^1$ using the north pole depicted as a black dot. The mean direction $\boldsymbol{\mu} \in \mathbb{S}_2^1$ is shown in orange.

Contrary to the previous case, the log-determinant of J_ψ alone is not enough to accommodate the density change when transforming from $\mathbb{R}^d$ to $\mathbb{S}_2^d$ [11]. The correct density ratio change is scaled by $\sqrt{\det J_\psi^\top J_\psi}$ instead, whose logarithm can be computed in $\mathcal{O}(d)$ time as

$$\log \sqrt{\det J_\psi^\top(\mathbf{z}) J_\psi(\mathbf{z})} = d \log \frac{2}{1 + \|\mathbf{z}\|^2} = d \log \rho_{\mathbf{z}},$$

with $\rho_{\mathbf{z}}$ given as stated above. The inverse function $\psi^{-1} : \mathbb{S}_2^d \subset \mathbb{R}^{d+1} \to \mathbb{R}^d$ is

$$\psi^{-1}(\mathbf{s}) = \mathbf{z} = \frac{[\mathbf{s}]_{1:d}}{1 - [\mathbf{s}]_{d+1}},$$

where $[\mathbf{s}]_{1:d}$ denotes the first d coordinates of $\mathbf{s}$ and $[\mathbf{s}]_{d+1}$ is the $(d+1)$-th coordinate of $\mathbf{s}$.

To interpolate points on the hypersphere, a spherical linear interpolation (*slerp*) [30] can be utilized. It is defined as follows. Let $\mathbf{s}_a$ and $\mathbf{s}_b$ be two unit vectors and $\omega = \cos^{-1}(\mathbf{s}_a^\top \mathbf{s}_b)$ be the angle between them. The interpolation path is then given by

$$\gamma(\lambda) = \frac{\sin((1 - \lambda)\omega)}{\sin(\omega)} \mathbf{s}_a + \frac{\sin(\lambda\omega)}{\sin(\omega)} \mathbf{s}_b, \ \ \text{for} \ \ \lambda \in [0, 1].$$

5   Experiments

We now evaluate the restriction of a normalizing flow to a unit p-norm sphere and compare them to a Gaussian base distribution. As we focus on a principled way of interpolating in flow-based generative models, we employ a fixed architecture per data set instead of aiming to achieve state-of-the-art density estimation. We use Glow [19] as the flow architecture for the experiments in the remainder of this

section. However, our approach is not limited to Glow, and the transformations and changes in the base distribution can also be used in other architectures. We also do not compare against other architectures as our contribution is a change of the base distribution, allowing for better interpolations.

5.1   Performance Metrics and Setup

Performance is measured in terms of *bits per dimension (BPD)*, calculated using $\log_2 p(\mathbf{x})$ divided by d; *Fréchet Inception Distance (FID) scores*, which have been shown to correlate highly with human judgment of visual quality [13]; and *Kernel Inception Distance (KID) scores* [3]. KID is similar to FID as it is based on Inception scores [29]. While the FID first fits a Gaussian distribution on the scores of a reference set and a set of interest and then compares the two distributions, the KID score is non-parametric, i.e., it does not assume any distribution and compares the Inception scores based on Maximum Mean Discrepancy (MMD). We follow previous work [3] and employ a polynomial kernel with degree three for our evaluations.

We measure bits per dimension on the test set and on interpolated samples. FID and KID scores are evaluated on generated and interpolated samples and then compared to a reference set, which is the training data. When generating data, we draw as many samples from the base distribution as we have for training. For interpolation, we focus on interpolation within classes and adopt regular linear interpolation for Gaussian-distributed samples, while using a spherical linear interpolation on the sphere for vMF-distributed samples. In this operation, we sample $n/5$ pairs of images from the training set and generate five equally spaced interpolated data instances per pair, resulting in n new images. From those interpolation paths, we only use the generated points and not the points which are part of training data. Hence, we are only considering previously unseen data.

We also compare against the norm-corrected linear interpolation (*nclerp*) defined in Equation (1). Note that a linearly spaced interpolation path is no longer linearly spaced after norm correction. The resulting interpolation paths are composed of images located closer to the endpoints and thus bias the evaluation. We include the results nevertheless for completeness.

The reported metrics are averages over three independent runs and include standard errors. The code is written in PyTorch [24]. All experiments run on an Intel Xeon CPU with 256GB of RAM using an NVIDIA V100 GPU.

5.2   Data

In our experiments, we utilize **MNIST** [22], **Kuzushiji-MNIST** [7], and **Fashion-MNIST** [34], which contain gray-scale images of handwritten digits, Hiragana symbols, and images of Zalando articles, respectively. All MNIST data sets consist of 60,000 training and 10,000 test images of size 28×28. In addition, we evaluate on **CIFAR10** [21], which contains natural images from ten classes. The data set has 50,000 training and 10,000 test images of size 32×32.

Table 1. Results for generative modeling averaged over three independent runs including standard errors.

	Base dist.	Test BPD	Sample FID	Sample KID
MNIST	Gaussian	1.59 ± 0.06	$\mathbf{34.53}\pm 0.83$	$\mathbf{0.033}\pm 0.001$
	vMF $\kappa = 1d$	$\mathbf{1.46}\pm 0.07$	40.07 ± 2.46	0.037 ± 0.001
	vMF $\kappa = 1.5d$	1.54 ± 0.09	40.39 ± 1.40	0.036 ± 0.001
	vMF $\kappa = 2d$	1.82 ± 0.08	39.82 ± 0.26	0.038 ± 0.001
	Dirichlet $\alpha = 2$	1.76 ± 0.12	40.08 ± 0.72	0.039 ± 0.001
K-MNIST	Gaussian	2.58 ± 0.11	35.34 ± 0.76	0.041 ± 0.001
	vMF $\kappa = 1d$	2.63 ± 0.06	36.63 ± 0.37	0.041 ± 0.001
	vMF $\kappa = 1.5d$	$\mathbf{2.48}\pm 0.06$	$\mathbf{35.00}\pm 0.61$	$\mathbf{0.040}\pm 0.001$
	vMF $\kappa = 2d$	2.51 ± 0.04	36.45 ± 0.42	0.041 ± 0.001
	Dirichlet $\alpha = 2$	2.50 ± 0.05	35.54 ± 0.39	$\mathbf{0.040}\pm 0.001$
F-MNIST	Gaussian	3.24 ± 0.04	66.64 ± 1.29	0.064 ± 0.003
	vMF $\kappa = 1d$	$\mathbf{3.16}\pm 0.03$	$\mathbf{60.45}\pm 3.34$	$\mathbf{0.055}\pm 0.005$
	vMF $\kappa = 1.5d$	3.30 ± 0.07	61.89 ± 1.29	0.056 ± 0.002
	vMF $\kappa = 2d$	3.22 ± 0.06	60.60 ± 3.47	$\mathbf{0.055}\pm 0.004$
CIFAR10	Gaussian	3.52 ± 0.01	71.34 ± 0.45	$\mathbf{0.066}\pm 0.001$
	vMF $\kappa = 1d$	3.43 ± 0.00	71.07 ± 0.78	0.069 ± 0.001
	vMF $\kappa = 1.5d$	$\mathbf{3.42}\pm 0.00$	$\mathbf{70.58}\pm 0.40$	0.068 ± 0.001
	vMF $\kappa = 2d$	$\mathbf{3.42}\pm 0.01$	71.00 ± 0.28	0.068 ± 0.001

5.3   Architecture

We employ the Adam optimizer [16] with a learning rate of 10^{-3}, clip gradients at 50, and use linear learning rate warm-up for the first ten epochs. Models were trained on MNIST data and CIFAR10 using mini-batches of size 256 and 128, respectively. All models are trained for 100 epochs without early stopping. We keep all architectures as close as possible to Glow, with the following deviations. For MNIST data, we use random channel permutations instead of invertible 1×1 convolutions. The number of filters in the convolutions of the affine coupling layers is 128. In Glow terms, we employ $L = 2$ levels of $K = 16$ steps each. For CIFAR10, our models have $L = 3$ levels of $K = 24$ steps each, while the affine coupling layers have convolutions with 512 filters. The architecture is kept the same across base distributions, except for the additional parameterless transformations to the restricted subspaces introduced in Section 4.

When comparing base distributions, we consider the following hyperparameters. For the vMF distribution, we use concentration values for which the partition function is finite. For consistency, the values we use are the same multiples of the data dimensionality d for each data set. The concentration values for the Dirichlet distribution are set to $\alpha = 2$, which refers to $2 \cdot \mathbf{1}^{d+1} \in \mathbb{R}^{d+1}$.

Table 2. Results for interpolation averaged over three independent runs including standard errors. Interpolations are in-class only and use five intermediate points; *lerp* refers to a linear interpolation; *nclerp* refers to the norm-corrected linear interpolation (Section 1) and *slerp* refers to the spherical interpolation.

	Base dist.	Type	BPD	FID	KID
MNIST	Gaussian	lerp	1.33 ± 0.05	5.10 ± 0.14	0.003 ± 0.000
	Gaussian	nclerp	1.44 ± 0.06	5.12 ± 0.30	0.003 ± 0.000
	vMF $\kappa = 1d$	slerp	**1.31** ± 0.09	**3.84** ± 0.36	**0.002** ± 0.000
	vMF $\kappa = 1.5d$	slerp	1.40 ± 0.10	4.22 ± 0.12	**0.002** ± 0.000
	vMF $\kappa = 2d$	slerp	1.63 ± 0.10	4.45 ± 0.06	**0.002** ± 0.000
	Dirichlet $\alpha = 2$	lerp	1.61 ± 0.10	5.81 ± 0.36	0.004 ± 0.001
K-MNIST	Gaussian	lerp	1.91 ± 0.17	19.71 ± 1.59	0.021 ± 0.002
	Gaussian	nclerp	2.15 ± 0.15	17.60 ± 1.48	0.020 ± 0.002
	vMF $\kappa = 1d$	slerp	2.08 ± 0.15	17.93 ± 3.72	0.020 ± 0.004
	vMF $\kappa = 1.5d$	slerp	**1.80** ± 0.07	22.72 ± 2.65	0.025 ± 0.003
	vMF $\kappa = 2d$	slerp	2.03 ± 0.14	**14.54** ± 2.51	**0.016** ± 0.003
	Dirichlet $\alpha = 2$	lerp	1.81 ± 0.04	24.09 ± 2.35	0.026 ± 0.003
F-MNIST	Gaussian	lerp	2.84 ± 0.10	13.06 ± 0.62	0.007 ± 0.001
	Gaussian	nclerp	2.93 ± 0.03	*7.80* ± 0.13	0.004 ± 0.000
	vMF $\kappa = 1d$	slerp	**2.66** ± 0.03	**12.16** ± 0.13	**0.006** ± 0.000
	vMF $\kappa = 1.5d$	slerp	2.84 ± 0.07	12.19 ± 1.07	**0.006** ± 0.001
	vMF $\kappa = 2d$	slerp	2.70 ± 0.05	15.11 ± 0.85	0.008 ± 0.001
CIFAR10	Gaussian	lerp	2.64 ± 0.06	58.63 ± 1.26	0.053 ± 0.001
	Gaussian	nclerp	3.32 ± 0.01	*14.29* ± 0.16	0.010 ± 0.000
	vMF $\kappa = 1d$	slerp	2.78 ± 0.05	**51.08** ± 0.37	0.010 ± 0.000
	vMF $\kappa = 1.5d$	slerp	2.66 ± 0.05	55.23 ± 5.14	0.047 ± 0.005
	vMF $\kappa = 2d$	slerp	**2.58** ± 0.08	52.65 ± 3.34	0.044 ± 0.004

5.4 Quantitative Results

We first evaluate the generative modeling aspects of all competitors. Table 1 summarizes the results in terms of bits per dimension on test data and FID and KID scores on generated samples for all data sets. Experiments with the Dirichlet base distribution were not successful on all data sets. The restrictions imposed to enable the use of the distribution demand a high numerical precision since every image on the simplex is represented as a non-negative vector that sums up to one. Consequently, we only report results on MNIST and Kuzushiji-MNIST. Using the vMF as a base distribution clearly outperforms the Gaussian in terms of bits per dimension on test data. As seen in the FID and KID scores, we perform competitive compared to the Gaussian for generating new data. Hence, the generative aspects of the proposed approach are either better or on par with the default choice of a Gaussian. Note that lower bits per dimension on test data and lower FID/KID scores on generated data might be obtained with more sophisticated models.

Fig. 6. Five interpolation paths of the norm-corrected linear interpolation (*nclerp*) depicting the problem of almost repeated endpoints (highlighted in blue) and thus a biased evaluation on CIFAR10.

We now evaluate the quality of interpolation paths generated via various approaches. Table 2 shows the results in terms of bits per dimension, FID, and KID scores for all data sets. The experiments confirm our hypothesis that an interpolation on a fixed-norm space yields better results as measured in bits per dimension, FID, and KID scores. The norm-corrected interpolation yields better FID and KID scores for Fashion-MNIST and CIFAR10. However, this heuristic produces interpolation paths that are biased towards the endpoints (cf. Figure 3) and hence are naturally closer to observed data, thus yield better FID and KID scores. This is depicted in Figure 6 where the first and last interpolant is very close to real data. More results on general interpolations within classes and across classes are provided in the supplementary material.

5.5   Qualitative Results

Figure 7 displays interpolation paths with five interpolants of four pairs of data from CIFAR10, created using the same architecture trained on different base distributions. We pick the best-performing model on BPD on test data from the multiple training runs for each base distribution. We visually compare a linear interpolation using a Gaussian base distribution against a spherical linear interpolation using a vMF base distribution with different concentration values. Naturally, the images in the center show the difference and the effects resulting from the choice of base distribution and, hence, the interpolation procedure.

Overall, the linear interpolation with a Gaussian tends to show mainly darker objects on brighter background (almost black and white images) in the middle of the interpolation path. This is not the case for the spherical interpolations using a vMF base distribution. Specifically, in the second example showing dogs, the checkerboard background of the left endpoint smoothly fades out for the vMF ($\kappa = 2d$) model while the Gaussian shows an almost white background. A similar effect happens in the last pair of images, highlighting the weaknesses of a
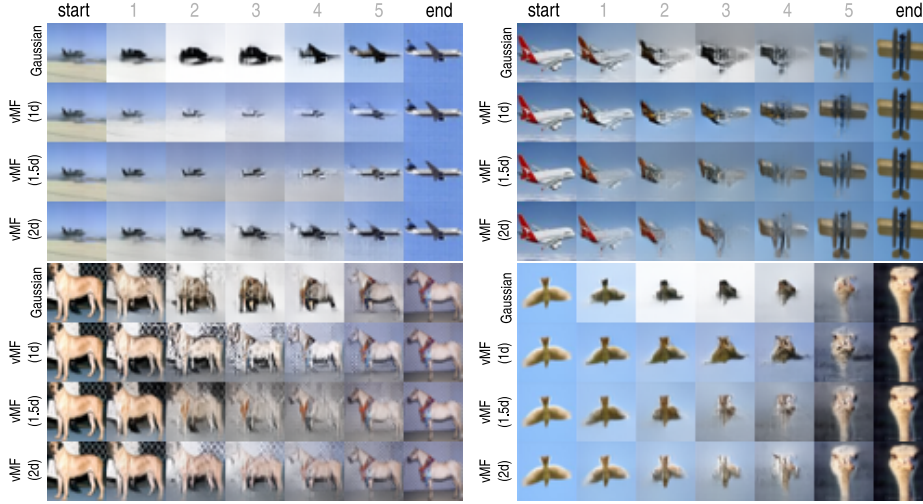
Fig. 7. Interpolation paths of four pairs of data from CIFAR10 using different models.
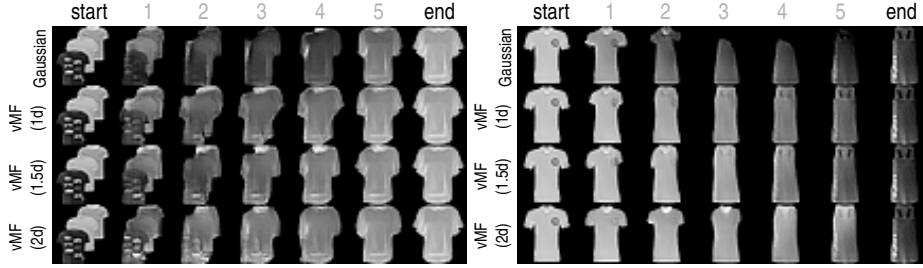

Fig. 8. Interpolation paths of two pairs from Fashion-MNIST using different models.

linear interpolation once again. By contrast, the vMF models generate images where those effects are either less prominent or non-existent, suggesting a path that strictly follows the data manifold. We provide more interpolation paths on CIFAR10 in the supplementary material.

Figure 8 depicts interpolation paths with five interpolants on two pairs of data from Fashion-MNIST. In both cases, the Gaussian model produces suboptimal images with visible color changes, which is not consistent with the endpoints. Furthermore, there is visible deformation of the clothing items.

6   Related Work

Interpolations are commonplace in generative modeling, being particularly useful for evaluating them. Spherical linear interpolations [30] are also proposed [33] to circumvent the problems depicted in Figure 1 in GANs and VAEs. However, as the Gaussian is kept as a base distribution, the difference in norms causes

problems similar to the norm corrected approach. The problem of interpolation is also investigated for GANs [1]. Specifically, they show that the quality of the generated images in the interpolation path improves when attempting to match the distribution of norms between interpolants and the GAN prior. The problem with the distribution mismatch while interpolating is also studied in [15].

Simultaneously learning a manifold and corresponding normalizing flow on it is also possible [5]. By contrast, in this paper, we employ a prescribed manifold, i.e., a p-norm sphere, on which the interpolation can be done in a principled way. Using a vMF distribution as a prior of VAEs is also used to encourage the model to learn better latent representations on data with hyperspherical structure [8, 35]. While results show improvements over a Gaussian prior, properties of our interest, such as interpolation, are not addressed.

Employing normalizing flows on non-Euclidean spaces, such as the hypersphere, was first proposed by [11]. They introduce a mapping for doing normalizing flows on hyperspherical data. The main difference from our setting is that the data is already on a sphere and is moved to $\mathbb{R}^d$, an unrestricted space, performing the entire flow in there instead, before moving back to the sphere. This avoids defining a flow on the sphere, which is studied in [27] for tori and spheres. Besides, normalizing flows on hyperbolic spaces are beneficial for graph-structured data [4].

A geometric analysis of autoencoders, showing that they learn latent spaces, which can be characterized by a Riemannian metric, is provided by [2]. With this, interpolations follow a geodesic path under this metric, leading to higher quality interpolations. Compared to our contribution, these approaches do not change the standard priors but propose alternative ways to interpolate samples. In contrast, we propose an orthogonal approach by changing the base distribution and imposing constraints on the representation in our training procedure. Consequently, standard interpolation procedures, such as the spherical linear interpolation, can be used in a principled way.

7   Conclusion

This paper highlighted the limitations of linear interpolation in flow-based generative models using a Gaussian base distribution. As a remedy, we proposed to focus on base representations with a fixed norm where the interpolation naturally overcomes those limitations and introduced normalizing flows onto unit p-norm spheres. Specifically, we showed for the cases $p \in \{1, 2\}$ that we could operate on the unit simplex and unit hypersphere, respectively. We introduced a computationally efficient way of using a Dirichlet distribution as a base distribution for the case of $p = 1$ and leveraged a von Mises-Fisher distribution using a stereographic projection onto a hypersphere for the case $p = 2$. Although the former suffered from numerical instabilities in a few experiments, our experimental results showed superior performance in terms of bits per dimension on test data and FID and KID scores on interpolation paths that resulted in natural transitions from one image to another. This was also confirmed by visually comparing interpolation paths on CIFAR10 and Fashion-MNIST.

Acknowledgements

References

1. Agustsson, E., Sage, A., Timofte, R., Gool, L.V.: Optimal transport maps for distribution preserving operations on latent spaces of generative models. In: International Conference on Learning Representations (2019)
2. Arvanitidis, G., Hansen, L.K., Hauberg, S.: Latent space oddity: on the curvature of deep generative models. In: International Conference on Learning Representations (2018)
3. Bikowski, M., Sutherland, D.J., Arbel, M., Gretton, A.: Demystifying MMD GANs. In: International Conference on Learning Representations (2018)
4. Bose, J., Smofsky, A., Liao, R., Panangaden, P., Hamilton, W.: Latent variable modelling with hyperbolic normalizing flows. In: International Conference on Machine Learning. pp. 1045–1055. PMLR (2020)
5. Brehmer, J., Cranmer, K.: Flows for simultaneous manifold learning and density estimation. Advances in Neural Information Processing Systems **33** (2020)
6. Chen, T.Q., Behrmann, J., Duvenaud, D.K., Jacobsen, J.H.: Residual flows for invertible generative modeling. In: Advances in Neural Information Processing Systems. pp. 9913–9923 (2019)
7. Clanuwat, T., Bober-Irizar, M., Kitamoto, A., Lamb, A., Yamamoto, K., Ha, D.: Deep learning for classical japanese literature. CoRR **abs/1812.01718** (2018)
8. Davidson, T.R., Falorsi, L., De Cao, N., Kipf, T., Tomczak, J.M.: Hyperspherical variational auto-encoders. In: Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence. pp. 856–865 (2018)
9. Dinh, L., Krueger, D., Bengio, Y.: NICE: non-linear independent components estimation. In: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings (2015)
10. Dinh, L., Shol-Dickstein, J., Bengio, S.: Density estimation using Real NVP. In: International Conference on Learning Representations (2017)
11. Gemici, M.C., Rezende, D., Mohamed, S.: Normalizing flows on riemannian manifolds. CoRR **abs/1611.02304** (2016)
12. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in Neural Information Processing Systems. pp. 2672–2680 (2014)
13. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: GANs trained by a two time-scale update rule converge to a local nash equilibrium. In: Advances in Neural Information Processing Systems. pp. 6626–6637 (2017)
14. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of GANs for improved quality, stability, and variation. In: International Conference on Learning Representations (2018)
15. Kilcher, Y., Lucchi, A., Hofmann, T.: Semantic interpolation in implicit models. In: International Conference on Learning Representations (2018)
16. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015)

17. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings (2014)
18. Kingma, D.P., Welling, M., et al.: An introduction to variational autoencoders. Foundations and Trends® in Machine Learning **12**(4), 307–392 (2019)
19. Kingma, D.P., Dhariwal, P.: Glow: Generative flow with invertible 1x1 convolutions. In: Advances in Neural Information Processing Systems. pp. 10215–10224 (2018)
20. Kingma, D.P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., Welling, M.: Improved variational inference with inverse autoregressive flow. In: Advances in Neural Information Processing Systems. pp. 4743–4751 (2016)
21. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images. Tech. rep. (2009)
22. LeCun, Y., Cortes, C., Burges, C.: Mnist handwritten digit database. ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist **2** (2010)
23. Papamakarios, G., Nalisnick, E., Rezende, D.J., Mohamed, S., Lakshminarayanan, B.: Normalizing flows for probabilistic modeling and inference. Journal of Machine Learning Research **22**(57), 1–64 (2021)
24. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems, pp. 8024–8035 (2019)
25. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. In: International Conference on Learning Representations, ICLR 2016 (2016)
26. Rezende, D., Mohamed, S.: Variational inference with normalizing flows. In: International Conference on Machine Learning. pp. 1530–1538. PMLR (2015)
27. Rezende, D.J., Papamakarios, G., Racaniere, S., Albergo, M., Kanwar, G., Shanahan, P., Cranmer, K.: Normalizing flows on tori and spheres. In: International Conference on Machine Learning. pp. 8083–8092 (2020)
28. Rippel, O., Adams, R.P.: High-dimensional probability estimation with deep density models. CoRR **abs/1302.5125** (2013)
29. Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training gans. In: Advances in neural information processing systems. pp. 2234–2242 (2016)
30. Shoemake, K.: Animating rotation with quaternion curves. In: Proceedings of the 12th annual conference on Computer graphics and interactive techniques. pp. 245–254 (1985)
31. Tabak, E.G., Turner, C.V.: A family of nonparametric density estimation algorithms. Communications on Pure and Applied Mathematics **66**(2), 145–164 (2013)
32. Tabak, E.G., Vanden-Eijnden, E.: Density estimation by dual ascent of the log-likelihood. Communications in Mathematical Sciences **8**(1), 217–233 (2010)
33. White, T.: Sampling generative networks. CoRR **abs/1609.04468** (2016)
34. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. CoRR **abs/1708.07747** (2017)
35. Xu, J., Durrett, G.: Spherical latent spaces for stable variational autoencoders. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. pp. 4503–4513 (2018)