A prior-based approximate latent Riemannian metric

Georgios Arvanitidis [1 2] **Bogdan Georgiev** [3] **Bernhard Schölkopf** [2]
[1]DTU Compute, Technical University of Denmark [2]MPI for Intelligent Systems, Tübingen [3]DeepMind

Abstract

Stochastic generative models enable us to capture the geometric structure of a data manifold lying in a high dimensional space through a Riemannian metric in the latent space. However, its practical use is rather limited mainly due to inevitable functionality problems and computational complexity. In this work we propose a surrogate conformal Riemannian metric in the latent space of a generative model that is simple, efficient and robust. This metric is based on a learnable prior that we propose to learn using a basic energy-based model. We theoretically analyze the behavior of the proposed metric and show that it is sensible to use in practice. We demonstrate experimentally the efficiency and robustness, as well as the behavior of the new approximate metric. Also, we show the applicability of the proposed methodology for data analysis in the life sciences.

1 Introduction

The *manifold hypothesis* states that in a high dimensional space the data has a low dimensional nonlinear geometric structure. One way to compute distances that respect this structure is by using discrete shortest paths on neighborhood graphs [Tenenbaum et al., 2000]. Nevertheless, this strategy does not allow to perform continuous analysis, as for example Riemannian statistics [Pennec, 2006]. Hence, methods based on latent variable models have been developed that enables us to compute continuous shortest paths.

Generative models provide a way to estimate the probability density of the given data lying in an ambient space $\mathcal{X}$. Most of the models utilize a latent space $\mathcal{Z}$,

Proceedings of the 25[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151.

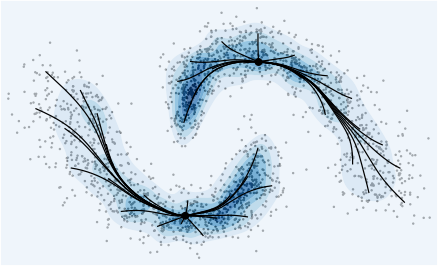


Figure 1: We propose a conformal Riemannian metric that is related to the learnable prior of a latent variable model. Intuitively, shortest paths prefer regions in the latent space $\mathcal{Z}$ with high density.

but the Variational Auto-Encoder (VAE) also learns a low dimensional representation of the data [Rezende et al., 2014; Kingma and Welling, 2014]. Unfortunately, the straight line distance in $\mathcal{Z}$ is misleading, and in addition, is not identifiable [Hauberg, 2018].

One solution is to compute shortest paths in $\mathcal{Z}$ using a Riemannian metric that is induced by the generator [Tosi et al., 2014; Arvanitidis et al., 2018]. This gives a natural and identifiable distance measure, since it is actually computed directly on the data manifold in $\mathcal{X}$. However, we need to estimate meaningfully the generator's uncertainty to properly capture the associated geometry in $\mathcal{Z}$. While this approach allows us to compute continuous and principled distances respecting the data manifold, it is not particularly efficient and robust. Specifically, the metric is based on the generator's Jacobian and its derivative, which is typically expensive to compute and complex. Also, meaningful uncertainty quantification is achieved using kernel methods [Arvanitidis et al., 2018], which limits further the robustness. These functionality problems motivate us to search for an approximate Riemannian metric to be used for practical purposes.

As regards VAEs, several improvements have been proposed and we are interested in learnable priors. Usually, in a VAE a simple prior over $\mathcal{Z}$ is chosen, as the unit Gaussian. This prior is not flexible and expressive enough in order to capture the structure of the

data representations, which potentially might be complex and multimodal. Therefore, learnable priors have been proposed that adapt to the distribution of the latent representations [Tomczak and Welling, 2018].

In this work, we propose to approximate the induced Riemannian metric in $\mathcal{Z}$ with a locally conformally flat surrogate metric that is based on a learnable prior. In particular, we first propose to utilize a basic energy-based model as a learnable prior in VAEs (Sec. 3.2). Then, we define a conformal Riemannian metric in $\mathcal{Z}$ that is inverse proportional to the prior (Sec. 4.1). This constitutes a robust metric that is also highly efficient both in computational speed, as well as in modeling capabilities. For example, in Fig. 1 we show that shortest paths in $\mathcal{Z}$ computed under our proposed metric respect the structure of the learned prior. Furthermore, we study theoretically when the proposed metric is a sensible approximation to the Riemannian metric that is induced by the generator. In the experiments, we compare the behavior of the two metrics, and we also show potential applications in life sciences.

2 Basics of Riemannian geometry

We consider Riemannian manifolds [do Carmo, 1992], which are smooth spaces where one can compute lengths between points. An intuitive way to think of a d-dimensional smooth manifold $\mathcal{M}$ is as an embedded smooth d-dimensional hypersurface in a higher dimensional Euclidean ambient space $\mathcal{X} = \mathbb{R}^D$, which locally is homeomorphic to a d-dimensional Euclidean space. In this perspective, one considers the tangent space $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ at a point $\mathbf{x} \in \mathcal{M}$ as a d-dimensional vector space in $\mathcal{X}$ that is tangential to the hypersurface at the point $\mathbf{x}$. Technically, a manifold is covered by a collection of *charts* (local parametrizations), and for simplicity we assume that a "sufficiently large" *global chart* exists (a global parametrization of the hypersurface). We denote this global chart by the mapping $h : \mathcal{H} \subseteq \mathbb{R}^d \to \mathcal{M} \subset \mathcal{X}$. By definition $h(\cdot)$ is a diffeomorphism onto its image and we say that $\mathcal{H}$ are the intrinsic coordinates of the manifold.

A *Riemannian metric* is a positive definite matrix that changes smoothly throughout the space. A smooth $\mathcal{M}$ together with a Riemannian metric constitutes a Riemannian manifold. Let an embedded $\mathcal{M} \subset \mathcal{X}$, the Riemannian metric $\mathbf{M}_{\mathcal{X}} : \mathcal{X} \to \mathbb{R}^{D\times D}_{\succ 0}$ defines a local inner product $\forall\, \mathbf{x} \in \mathcal{M}$ on the tangent space $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ between two tangent vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^D$ as $\langle \mathbf{u}, \mathbf{v}\rangle_{\mathbf{x}} = \langle \mathbf{u}, \mathbf{M}_{\mathcal{X}}(\mathbf{x})\mathbf{v}\rangle$. The global chart $h(\cdot)$ allows to map a vector $\overline{\mathbf{v}} \in \mathcal{H}$ to a unique tangent vector $\mathbf{v} \in \mathcal{T}_{\mathbf{x}}\mathcal{M}$ using the Jacobian matrix $\mathbf{J}_h : \mathcal{H} \to \mathbb{R}^{D\times d}$ as $\mathbf{v} = \mathbf{J}_h(\mathbf{z})\overline{\mathbf{v}}$. For smooth $h(\cdot)$, a Riemannian metric $\mathbf{M}_{\mathcal{H}} : \mathcal{H} \to \mathbb{R}^{d\times d}_{\succ 0}$ is induced in $\mathcal{H}$ as $\mathbf{M}_{\mathcal{H}}(\cdot) = \mathbf{J}_h(\cdot)^\intercal \mathbf{M}_{\mathcal{X}}(h(\cdot))\mathbf{J}_h(\cdot)$.

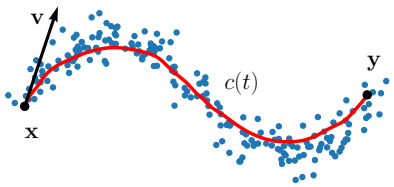


Figure 2: A shortest path $c(t)$ and a tangent vector $\mathbf{v}$.

In most cases, we consider $\mathcal{X}$ as Euclidean space so $\mathbf{M}_{\mathcal{X}}(\cdot) = \mathbb{I}_D$, which implies that the metric $\mathbf{M}_{\mathcal{H}}(\cdot)$ in $\mathcal{H}$ is induced by the embedding [Arvanitidis et al., 2019].

A Riemannian metric shows how the distances change in an infinitesimal region and enables us to compute the length of a curve $\gamma : [0, 1] \to \mathcal{M} \subset \mathcal{X}$ with $\int_0^1 \sqrt{\langle \dot{\gamma}(t), \dot{\gamma}(t)\rangle_{\gamma(t)}}dt = \int_0^1 \sqrt{\langle \dot{c}(t), \mathbf{M}_{\mathcal{H}}(c(t))\dot{c}(t)\rangle}dt$, where $\gamma(t) = h(c(t))$ and $\dot{\gamma}(t) = \partial_t\gamma(t) = \mathbf{J}_h(c(t))\dot{c}(t) \in \mathcal{T}_{\gamma(t)}\mathcal{M}$ the velocity of the curve. We can compute the length of $\gamma(t)$ in intrinsic coordinates $c(t) \in \mathcal{H}$ using the metric $\mathbf{M}_{\mathcal{H}}(\cdot)$. Also, we find the shortest path between two points in $\mathcal{H}$ by applying the Euler-Lagrange equations at the curve energy. This gives a system of second order non-linear ordinary differential equations (ODEs) $\ddot{c}(t) = F(\dot{c}(t), c(t), t)$ which can be found in App. A. Intuitively, the shortest paths are pulled towards areas of $\mathcal{H}$ where $|\mathbf{M}_{\mathcal{H}}(\cdot)|$ is small. These curves are known as *geodesics*. Additional details in App. A.

A Riemannian metric $\mathbf{M}(\cdot)$ is conformal to $\widetilde{\mathbf{M}}(\cdot)$ when for a positive smooth function $m : \mathcal{H} \to \mathbb{R}_{>0}$ we have $\widetilde{\mathbf{M}}(\mathbf{z}) = m(\mathbf{z})\mathbf{M}(\mathbf{z})$. The conformal metric is simply a scaling of $\mathbf{M}(\cdot)$, while the simplest example is to consider $\widetilde{\mathbf{M}}(\mathbf{z}) = m(\mathbf{z})\mathbb{I}_d$. This metric has some appealing properties as interpretability and efficiency. More specifically, the corresponding ODEs system for computing a shortest path simplifies to

$$\ddot{c}(t) = \frac{\nabla m(c(t))\dot{c}(t)^\intercal \dot{c}(t) - 2\dot{c}(t)\nabla m(c(t))^\intercal \dot{c}(t)}{2\; m(c(t))}, \quad (1)$$

where $\nabla m : \mathcal{H} \to \mathbb{R}^d$ the gradient of $m(\cdot)$. The interpretability implies that we can easily control the shortest paths behavior in $\mathcal{H}$ by designing $m(\cdot)$ accordingly.

The *manifold hypothesis* is that the data lie uniformly near an embedded $\mathcal{M} \subset \mathcal{X}$. However, for a given set of finite noisy observations a global chart $h(\cdot)$ rarely exists and d is unknown. Hence, a practical method to capture the geometry of $\mathcal{M}$ is to learn a function $g : \mathcal{Z} \subseteq \mathbb{R}^{d'} \to \mathcal{X}$ to approximate the data, which is a smooth immersion but not constrained to be a diffeomorphism as $h(\cdot)$, while in general $\mathcal{Z} \neq \mathcal{H}$. We then use $\mathbf{J}_g(\cdot)$ to compute $\mathbf{M}_{\mathcal{Z}}(\cdot)$, for which a desired meaningful behavior is to be small in parts of $\mathcal{Z}$ that correspond to regions of $\mathcal{M}$ with non-zero data density. This is known as the *pull-back* metric, and in practice, we learn $g(\cdot)$ using generative models (see Sec. 4).

Riemannian metric learning. Apart from the pull-back metric, we can also learn a Riemannian metric directly in $\mathcal{X}$ from the given data $\mathbf{x}_{1:N} \in \mathcal{X}$, using a parametric metric $\mathbf{M}_\lambda : \mathcal{X} \to \mathbb{R}^{D\times D}_{\succ 0}$ and estimating the optimal parameters λ. Actually, this metric changes the way we measure distances in $\mathcal{X}$ in order to respect the data manifold structure. So the $|\mathbf{M}_\lambda(\cdot)|$ should be small in regions of $\mathcal{X}$ with non-zero data density, such that to pull the shortest paths towards them (Fig. 2).

One approach is to consider a predefined set of metrics centered at base points in $\mathcal{X}$, and then using a kernel to compute the Riemannian metric as their weighted sum [Hauberg et al., 2012]. In a similar spirit, Arvanitidis et al. [2016] used a kernel to compute the Riemannian metric as the inverse local diagonal covariance matrix of the data. Also, Arvanitidis et al. [2019] proposed a simple method to construct conformal Riemannian metrics directly from the data by multiplying a positive function in $\mathcal{X}$ with the Euclidean metric. For details about these metrics see App. B. However, in these approaches the parameters λ of the metric are typically fixed, and in general, it is a challenging task to find the best λ [Arvanitidis et al., 2017].

In contrast, Lebanon [2002] proposed to optimize λ using the density $p_\lambda(\mathbf{x}) \propto (\sqrt{|\mathbf{M}_\lambda(\mathbf{x})|})^{-1}$ and under the assumption of independent and identically distributed data to maximize the likelihood $\prod_{n=1}^{N} (\sqrt{|\mathbf{M}_\lambda(\mathbf{x}_n)|})^{-1} / \int_{\mathcal{X}} (\sqrt{|\mathbf{M}_\lambda(\mathbf{x}')|})^{-1} d\mathbf{x}'$. Hence, the density should be high near the given data, which means that the metric should be small, while the regularizer prevents the metric of becoming zero. The quantity $\sqrt{|\mathbf{M}_\lambda(\mathbf{x})|}$ is the *magnitude*, which is a scaling factor for the Lebesgue $d\mathbf{x}$ and represents the local distrortion of the distance.

This Riemannian metric learning approach has the following disadvantages. We have to define explicitly the parametric form of the metric before the training, which potentially limits its flexibility. Also, in higher dimensions it is hard to guarantee the actual behavior and usability of such an ad-hoc metric. In addition, the optimization is challenging especially in high dimensions due to the normalization constant. Therefore, this methodology is mostly limited to low dimensional spaces and rather simple data manifolds, but the actual formulation motivates us to relate a density function with a conformal metric (see Sec. 4).

3 Generative Models

An efficient way to approximate the underlying probability density of the observations $\mathbf{x}_{1:N} \in \mathcal{X}$ is with a deep generative model and recent advances exhibit great performance. There are several types of generative models such as Variational Auto-Encoders (VAEs) [Kingma and Welling, 2014; Rezende et al., 2014], Generative Adversarial Networks (GANs) [Goodfellow et al., 2014] and flow based models [Dinh et al., 2016]. We consider the VAE that utilizes a low dimensional latent space $\mathcal{Z}$ where we can represent the given data.

Specifically, we use a likelihood function $p_\theta(\mathbf{x}|\mathbf{z})$ that is typically chosen to be a Gaussian $\mathcal{N}(\mathbf{x}|\mu_\theta(\mathbf{z}), \mathbb{I}_D \sigma^2_\theta(\mathbf{z}))$ and a prior $p(\mathbf{z})$ as the $\mathcal{N}(0, \mathbb{I}_d)$ over the latent variables. We parametrize the likelihood with deep neural networks $\mu_\theta : \mathcal{Z} \to \mathcal{X}$ and $\sigma^2_\theta : \mathcal{Z} \to \mathbb{R}^D_{>0}$. Also, we use an approximate posterior $q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}|\mu_\phi(\mathbf{x}), \mathbb{I}_d \sigma^2_\phi(\mathbf{x}))$, where again $\mu_\phi : \mathcal{X} \to \mathcal{Z}$ and $\sigma^2_\phi : \mathcal{X} \to \mathbb{R}^d_{>0}$ are deep neural networks. Then, using Jensen's inequality we derive the evidence lower bound (ELBO)

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \mathrm{KL}[q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})], \qquad (2)$$

which is a lower bound to the data log-likelihood. We optimize θ, ϕ using objective and the reparametrization trick $\mathbf{z} = \mu_\phi(\mathbf{x}) + \mathrm{diag}(\varepsilon) \cdot \sigma_\phi(\mathbf{x})$, where $\varepsilon \sim \mathcal{N}(0, \mathbb{I}_d)$, which allows to compute stochastic gradients with low variance [Mohamed et al., 2020].

3.1 Prior learning in VAEs

One variant of the standard VAE is to replace the simple prior with a learnable one. Intuitively, the learned prior captures the structure of the latent representations in $\mathcal{Z}$, which is similar to the behavior of a pull-back Riemannian metric.

One of the first successful methods to learn the prior in a VAE is the VampPrior [Tomczak and Welling, 2018]. In this approach the learnable prior is chosen to be the aggregated posterior $p(\mathbf{z}) \triangleq q(\mathbf{z}) = \int_{\mathcal{X}} q(\mathbf{z}|\mathbf{x}) p(\mathbf{x}) d\mathbf{x}$, where $p(\mathbf{x})$ is the true data density. In a standard VAE this is essentially a huge Gaussian mixture model, since typically we approximate this integral using the training data $p(\mathbf{z}) \approx \frac{1}{N} \sum_{n=1}^{N} q(\mathbf{z}|\mathbf{x}_n)$. Of course, such a prior can easily overfit, so the authors proposed to use instead only K learnable inducing points $\mathbf{x}_{1:K}$. This simple prior is empirically shown to be very effective, but when the data dimension is high, training the inducing points is computationally expensive. Also, is hard to chose the number K of inducing points.

Another set of approaches is related to energy-based models. Bauer and Mnih [2019] learns a function to accept or reject samples from a base prior as the unit Gaussian. Pang et al. [2020] learns an energy-based model prior by optimizing the log-likelihood of the data requiring iterative expensive Markov Chain Monte Carlo sampling in $\mathcal{Z}$ for the prior and the true posterior. Aneja et al. [2020] learns post-hoc an energy-based model prior by contrasting samples from the aggregated posterior to samples from a base prior. These approaches motivate our proposed prior.

3.2 Our learnable prior for VAEs

Let a deep neural network $f_\psi : \mathcal{Z} \to \mathbb{R}$ and a base distribution $p(\mathbf{z}) = \mathcal{N}(0, \mathbb{I}_d)$. We use as learnable prior the energy-based model [LeCun et al., 2006]

$$\nu_\psi(\mathbf{z}) = \exp(f_\psi(\mathbf{z}))p(\mathbf{z}) \;/\; \mathcal{C}, \tag{3}$$

where $\mathcal{C} = \int_{\mathcal{Z}} \exp(f_\psi(\mathbf{z}))p(\mathbf{z})d\mathbf{z}$ is the normalization constant. We plug this prior in the evidence lower bound of the VAE (Eq. 2) and the ELBO becomes

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \mathrm{KL}[q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})] + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[f_\psi(\mathbf{z})] - \log(\mathcal{C}), \tag{4}$$

which can be optimized using stochastic gradients as well. The challenge in this optimization problem is to estimate efficiently the term with the normalization constant. However, since the dimensionality of $\mathcal{Z}$ is usually low this allows us to estimate the normalization constant $\mathcal{C}$ relying on basic Monte Carlo as $\mathcal{C} = \int_{\mathcal{Z}} \exp(f_\psi(\mathbf{z}))p(\mathbf{z})d\mathbf{z} \approx \frac{1}{S}\sum_{s=1}^{S} \exp(f_\psi(\mathbf{z}_s))$ where $\mathbf{z}_s \sim p(\mathbf{z})$. Nevertheless, more sophisticated techniques for estimating the constant can be used.

Even if this is a rather simple prior, it comes with some desirable properties. First, the behavior is easy to interpret, as the prior increases near the latent codes of the data, while in contrast, the normalization constant tries to reduce its value in regions of $\mathcal{Z}$ with no codes. This implicit regularization does not allow the model to overfit the latent codes, which is directly related to the effectiveness of the integration. Also, contrastive techniques can be used in order to control even further the prior fitting i.e. far from latent codes to push the prior towards zero. In addition, the KL divergence of the standard VAE still appears in the objective. This is beneficial because the encoder is still encouraged to provide a meaningful structure for the latent codes, while in a different case the representations could be placed sparsely without any structure in $\mathcal{Z}$ depending on the flexibility of $f_\psi(\cdot)$.

Clearly, our proposed energy-based model prior is a rather simple choice, while being closely related to more advanced models which aim to improve generative modeling [Pang et al., 2020; Aneja et al., 2020]. However, to the best of our knowledge, such a prior has not be used in the standard VAE setting. In addition, our main motivation for proposing this prior is not to improve the generative modeling performance, but instead to have a flexible prior that adapts to the latent representations, which is efficient to evaluate and derivate. This prior is the base to define a conformal metric in $\mathcal{Z}$ (see Sec. 4.1), which approximates the geometry of the data manifold, while being at the same time efficient and robust.

4 Riemannian metric learning via generative modeling

Instead of learning a parametric Riemannian metric in $\mathcal{X}$ from data (see Sec. 2), we discuss how to learn one in the latent space $\mathcal{Z}$ of a generative model. Briefly, a generator $g : \mathcal{Z} \to \mathcal{X}$ induces a pull-back metric in $\mathcal{Z}$ (see Sec. 2) that informs us about the local distortions of $\mathcal{Z}$ when mapping through $g(\cdot)$. This metric captures the geometry of the data manifold lying in $\mathcal{X}$. However, as we discuss below even if this is a theoretically rigorous approach, it has practical disadvantages.

Tosi et al. [2014] first proposed to capture the geometry of a data manifold by modeling the generator $g(\cdot)$ using a Gaussian Process Latent Variable Model (GP-LVM) [Lawrence, 2005]. Here, the generator is taken to be a Gaussian process $g \sim \mathrm{GP}(0, k(\mathbf{z}, \mathbf{z}'))$ and the latent codes of the data $\mathbf{z}_{1:N}$ are trained. Since GPs are closed under differentiation the Jacobian $\mathbf{J}_g(\cdot)$ is a random process, which induces a stochastic Riemannian metric in $\mathcal{Z}$. This metric comes with a meaningful behavior, since it is small near the latent codes and increases when the uncertainty of $g(\cdot)$ increases. This properly captures in $\mathcal{Z}$ the geometry of the data manifold, but it is not very useful due to the practical constraints of the GP.

In a similar spirit, Arvanitidis et al. [2018] showed that a Riemannian metric is naturally induced in the latent space of deep generative models. The generator of a standard VAE $\mathbf{x} = g(\mathbf{z}) = \mu_\theta(\mathbf{z}) + \mathrm{diag}(\varepsilon) \cdot \sigma_\theta(\mathbf{z})$ with $\varepsilon \sim \mathcal{N}(0, \mathbb{I}_D)$ is a stochastic function, which induces a random Riemannian metric in $\mathcal{Z}$ with expectation

$$\mathbf{M}_\theta(\mathbf{z}) = \mathbf{J}_{\mu_\theta}(\mathbf{z})^\intercal \mathbf{J}_{\mu_\theta}(\mathbf{z}) + \mathbf{J}_{\sigma_\theta}(\mathbf{z})^\intercal \mathbf{J}_{\sigma_\theta}(\mathbf{z}). \tag{5}$$

When the uncertainty $\sigma_\theta(\cdot)$ of $g(\cdot)$ increases, the second term of the metric increases, which constitutes a meaningful behavior. However, $\mu_\theta(\cdot)$ and $\sigma_\theta(\cdot)$ are usually parametrized as deep neural networks that are known to extrapolate arbitrarily. A solution proposed by Arvanitidis et al. [2018] is to model the precision $\xi_\theta(\mathbf{z}) = (\sigma^2_\theta(\mathbf{z}))^{-1}$ using a positive Radial Basis Function (RBF) network. Hence, moving further from the latent codes decreases the precision, which directly makes the second term of the expected metric Eq. 5 to increase. Therefore, a stochastic generator with meaningful uncertainty estimation enables us to properly capture the geometry of the data manifold in $\mathcal{Z}$ [Hauberg, 2018]. Moreover, it has been theoretically shown by Eklund and Hauberg [2019] that it is sensible to use this expected metric.

Nevertheless, even if this metric enables us computing shortest paths in $\mathcal{Z}$ that respect the latent structure, it comes with practical drawbacks. In particular, we need to set the parameters of the RBF network as the

number of components K and the bandwidth for a Gaussian kernel, which is in general a difficult problem. In addition, to find one curve we evaluate the ODE system several times, which involves the metric and its derivative that are based on the Jacobian of $g(\cdot)$. For complex generators this is computationally very expensive. Also, by definition $g(\cdot)$ has to be twice differentiable, so we cannot use complicated architectures. Finally, the ODE system becomes highly unstable affecting negatively the performance of the solvers [Arvanitidis et al., 2019]. Even solvers that minimize the curve's energy using automatic differentiation are still slow for complex models [Yang et al., 2018].

Stochastic generators provide one way to properly capture in $\mathcal{Z}$ the geometry of the data manifold lying in $\mathcal{X}$. Moreover, this approach enables us to derive more informative metrics by considering the space $\mathcal{X}$ as a Riemannian manifold [Arvanitidis et al., 2020]. However, due to the practical disadvantages, we are interested in approximating the geometry in $\mathcal{Z}$ using a simple, efficient and robust surrogate Riemannian metric.

4.1 Our prior-based conformal metric

We propose a new Riemannian metric in $\mathcal{Z}$ that approximates the behavior of the true pull-back metric $\mathbf{M}_\theta(\cdot)$ (see Eq. 5), while having several advantages as regards its practicality. Let a VAE with a trainable smooth prior $\nu_\psi(\mathbf{z})$ for which we can evaluate easily the density function and its derivative. We are motivated by Lebanon [2002] where a probability density is defined to be inverse proportional to the magnification factor (see Sec. 2). In a similar spirit, we propose an approximation to the true $\mathbf{M}_\theta(\cdot)$ in $\mathcal{Z}$ using the following locally conformally flat Riemannian metric

$$\mathbf{M}_\psi(\mathbf{z}) = m(\mathbf{z}) \cdot \mathbb{I}_d = (\alpha \cdot \nu_\psi(\mathbf{z}) + \beta)^{-2/d} \cdot \mathbb{I}_d, \qquad (6)$$

where $\alpha,\ \beta > 0$ are scaling constants to lower and upper bound the metric, respectively. This metric by definition is conformal to the Euclidean metric $\mathbb{I}_d$ in $\mathcal{Z}$, and also, the quantity $\sqrt{|\mathbf{M}_\psi(\mathbf{z})|} = (\alpha \cdot \nu_\psi(\mathbf{z}) + \beta)^{-1}$ is inverse proportional to the learnable prior.

The metric $\mathbf{M}_\psi(\cdot)$ has an interepretable and meaningful behavior, as in regions of $\mathcal{Z}$ where the density is high the metric is small, and thus, the shortest paths are pulled towards the latent codes. Intuitively, this properly captures the geometry of the high dimensional data manifold, at least in the sense that paths tend to avoid regions of $\mathcal{Z}$ with no latent codes. Additionally, the metric is directly learned from the data, while depending on the flexibility of $\nu_\psi(\cdot)$ it can be highly adaptive. This further implies that the metric is more robust in higher dimensional latent spaces, as in principle, does not depend on a predefined parametric form and/or a kernel. Also, as a conformal metric the corresponding ODEs system simplifies (see Eq. 1). Hence, the proposed prior Eq. 3 seems to be a perfect choice, since it is flexible, adaptive and efficient to evaluate, as well as to derivate.

In order to prevent the prior from overfitting the latent codes we can control the capacity of $f_\psi(\cdot)$. Also, we can improve the prior fitting using training techniques as contrastive learning. Note that our prior in Eq. 6 can be easily replaced by a more sophisticated model that performs better, as long as the functional form of the density and its derivative are easy to compute.

4.2 Analysis of our proposed metric

We analyze the behavior of the proposed metric that is seemingly a sensible approximation to the pull-back metric of Arvanitidis et al. [2018]. Let the smooth manifold $\mathcal{Z} = \mathbb{I}_d$ and two Riemannian metrics, the pull-back $\mathbf{M}_\theta(\cdot)$ and the proposed conformal $\mathbf{M}_\psi(\cdot)$ metric. We are interested in whether these two metrics result in equivalent shortest paths, meaning that the corresponding curves should be similar. We study theoretically and compare the behavior in three specific cases, while we provide the respective constructive examples in the experiments (see Fig. 3). All proofs can be found in App. D. Based on the manifold hypothesis, we assume that the distribution $p(\mathbf{x})$ of the data near $\mathcal{M}$ is approximately uniform, and in addition, we assume that the learned $g(\cdot)$ together with our proposed prior estimates this density sufficiently well.

Proposition 1. *Let a learned $g(\cdot)$ with $\sigma^2_\theta(\cdot)$ an inverse RBF network and $\nu_\psi(\mathbf{z})$ the proposed prior. Then, the magnitude of the metrics $\mathbf{M}_\theta(\cdot)$ and $\mathbf{M}_\psi(\cdot)$ is maximum in the same region of $\mathcal{Z}$ where $\nu_\psi(\mathbf{z}) \to 0$.*

Actually, Prop. 1 means that the two metrics capture the same latent structure, since for both the magnitude becomes high as we move far from the latent codes. This directly implies that the global behavior of the shortest paths tends to be similar, which is useful in practice. In particular, the paths are pulled towards the regions where $\nu_\psi(\mathbf{z}) > 0$ avoiding regions with high metric magnitude, which is a desirable behavior.

However, by definition the two metrics are structurally different as $\mathbf{M}_\theta(\cdot)$ is a full metric tensor, while $\mathbf{M}_\psi(\cdot)$ is a conformal metric. Consequently, the local behavior of the paths is rather hard to perfectly match. Also, we show below that the under some conditions the behavior of the metrics is exactly the opposite.

Proposition 2. *Let a neighborhood $\mathcal{U} \subset \mathcal{M}$ where $p(\mathbf{x})$ is uniform, a learned $g(\cdot)$ and the proposed prior $\nu_\psi(\mathbf{z})$. If $\sigma^2_\theta(\cdot) \approx \varepsilon$, where $\varepsilon > 0$ a small scalar, then in the corresponding region in $\mathcal{Z}$ the two metrics are related as $\sqrt{|\mathbf{M}_\theta(\cdot)|} = \sqrt{|\mathbf{M}_\psi(\cdot)|}^{-1}$.*

Therefore, Prop. 2 implies that the associated shortest paths locally can be different. Nevertheless, another factor that influences the paths is the curvature. Intuitively, this quantity represents for a metric the rate of change, and in general, the paths tend to avoid regions with high curvature. When in a neighborhood the curvature is low the magnitude of the metric does not deviate much. So even if locally the magnitudes are inverse proportional, as long as the curvature is low, the behavior of the paths can be similar.

Proposition 3. *Let a neighborhood $\mathcal{U} \subset \mathcal{M}$ where $p(\mathbf{x})$ is uniform. We assume that in the corresponding region in $\mathcal{Z}$ the generator's uncertainty $\sigma^2_\theta(\cdot) \approx \varepsilon$, where $\varepsilon > 0$ a small scalar and that the $\mu_\theta(\cdot)$ has low curvature. Then, for both metrics $\mathbf{M}_\theta(\cdot)$ and $\mathbf{M}_\psi(\cdot)$ the shortest paths within this region are straight lines.*

Of course, in practice it is rather hard to control effectively the curvature of the generator. Since this depends on several factors as the given data manifold, the modeling choices and the training. Additionally, our assumptions in general do not hold for data manifolds, which complicates the analysis. In particular, the uncertainty $\sigma^2_\theta(\cdot)$ is not constant, the distribution $p(\mathbf{x})$ is not uniform, while the intrinsic dimensionality of the data manifold may change. We focus on the last case, as it has direct implications in practice.

Proposition 4. *Let a data manifold $\mathcal{M}$ where $p(\mathbf{x})$ is uniform. We assume that there is a neighborhood $\mathcal{U} \subset \mathcal{M}$ where the intrinsic dimensionality increases. Then, in the corresponding region of $\mathcal{Z}$ the two metrics $\mathbf{M}_\theta(\cdot)$, $\mathbf{M}_\psi(\cdot)$ have exactly the opposite behavior.*

Nevertheless, due to Prop. 1 we argue that the proposed metric is a sensible approximation to the true pull-back metric. In particular, we showed that both metrics capture the same latent structure, and hence, the associated shortest paths globally are similar. In contrast, we presented some cases where the behavior of the paths locally differs, and we note that potentially additional problematic cases might exist. Also, we propose a conformal metric, which has a rather limited expressivity compared to the true pull-back. However, we argue that this type of problems are negligible in practice, as long as the shortest paths for both metrics are similar and can be computed efficiently.

Note that we can easily take our metric into account during the training as it is based on the learnable prior. In this way, we can influence the generative model by considering interpretable inductive biases through geometric formulations. For example, we can consider pointwise regularizers of the form $||\mathbf{M}_\psi(\mathbf{z}) - \mathbf{M}_\theta(\mathbf{z})||^2_F$. Of course, this specific regularizer premises that $\mathbf{M}_\theta(\cdot)$ is robust and efficient, which is not currently the case, so we do not use this in our experiments.

	Standard	VampPrior	Ours
MNIST	85.38	83.28	83.56
FMNIST	227.12	224.15	224.53
MNIST (100)	95.51	90.24	91.70
FMNIST (100)	87.17	81.83	84.06

Table 1: The mean NLL on test data.

5 Experiments

Our experimental setting is two fold. First, we compare our prior to the state-of-the-art VampPrior Tomczak and Welling [2018]. Note that our goal is not to improve generative modeling, but to show that $\nu_\psi(\cdot)$ adapts well to the latent codes, so it is a sensible choice for $\mathbf{M}_\psi(\cdot)$. Then, we compare the proposed metric with the pull-back $\mathbf{M}_\theta(\cdot)$ of Arvanitidis et al. [2018] on several aspects as the robustness and the efficiency of shortest paths. Also, we provide a constructive example based on the analysis of Sec. 4.2. Finally, we show potential applications of Riemannian statistics in life sciences. Further experimental details in App. E.

5.1 Performance of the proposed prior

We compare in terms of negative log-likelihood (NLL) our learnable prior to the standard unit Gaussian and the VampPrior. We train 10 Conv-VAEs on MNIST and FashionMNIST datasets and we report the mean NLL of test data in Table 1, which we computed using importance sampling with 5000 samples as $p(\mathbf{x}) \approx \frac{1}{S}\sum_{s=1}^{S} \frac{p_\theta(\mathbf{x}|\mathbf{z}_s)p(\mathbf{z}_s)}{q_\phi(\mathbf{z}_s|\mathbf{x})}$ where $\mathbf{z}_s \sim q_\phi(\mathbf{z}|\mathbf{x})$. In addition, using PCA, we projected the datasets in 100 dimensions and we fitted 10 VAEs with Gaussian decoders. This already captures $> 90\%$ of the data variance, while enables us to use stochastic decoders such that to use the pull-back metric in the latent space. In both cases the dimension of the latent space is $d = 10$.

We use for the VampPrior $K = 500$ learnable inducing points and for our prior $f_\psi(\cdot)$ a fully connected 2-layer deep network with 128 units per layer and `tanh` activations. The results in Table 1 show that our proposed prior is comparable to VampPrior, while being always better than the unit Gaussian prior. This shows that $\nu_\psi(\cdot)$ adapts well to the latent codes during training.

5.2 Comparing the behavior of the metrics

We provide empirical evaluation for the analysis in Sec. 4.2. We generate a data manifold in $\mathcal{X} = \mathbb{R}^3$ as $[\mathbf{z}, 0.25 \cdot \sin(z_1)] + \varepsilon$ where $z_j \sim \mathcal{U}(0, 2\pi)$, $j = 1, 2$ and $\varepsilon \sim \mathcal{N}(0, 0.1^2 \cdot \mathbb{I}_3)$, with a hole in the center, and with a ball of uniformly sampled points in the center. We trained a VAE per dataset with our proposed

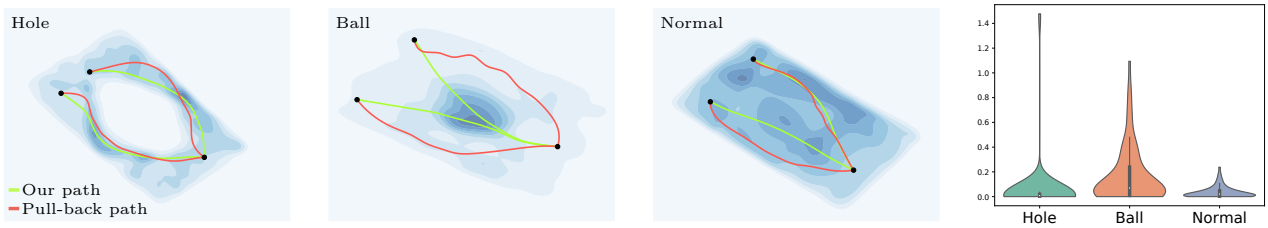


Figure 3: We demonstrate the three cases analyzed in Sec. 3. *Left*: The two metrics behave similarly since the uncertainty estimation aligns well with the prior (Prop. 1). *Middle*: The behavior of the metrics is exactly the opposite, as in the area with high prior density the generator has high uncertainty (Prop. 4). *Right*: We expect the shortest paths locally to be similar, as long as the prior is uniform and the curvature of the generator is small (Prop. 3). In addition, we show the distribution of distances between the curves for each case respectively. Note that in the *hole* case the paths tend to be similar except a few outliers, which constitutes a useful behavior for practical purposes in which we are interested, while in the problematic *ball* case many paths are not similar.

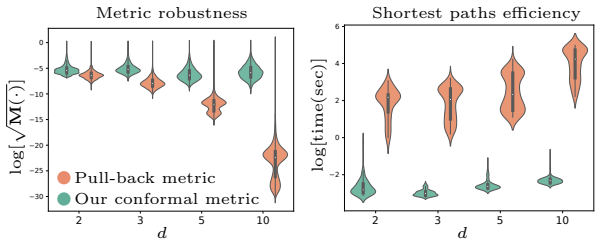


Figure 4: *Left*: The magnitude of $\mathbf{M}_\psi(\cdot)$ remains stable in higher dimensions, while $\mathbf{M}_\theta(\cdot)$ due to the RBF is not robust. In fact, some latent codes fall far from the RBF centers, so the second term of $\mathbf{M}_\theta(\cdot)$ becomes large overestimating curve lengths. *Right*: The $\mathbf{M}_\theta(\cdot)$ results in a complex and unstable ODEs system that limits the efficiency of the solver and also leads it many times to failure. While with our metric $\mathbf{M}_\psi(\cdot)$ the system is significantly easier to solve.

prior and we fitted post-hoc an RBF network for each to get the pull-back metric. In Fig. 3 we show the $\mathcal{Z} = \mathbb{R}^2$. We define the distance between two curves as $\int_0^1 ||c_1(t) - c_2(t)||_2^2 dt$, where each curve is parametrized with unit speed under the Euclidean metric. This makes curves coming from different Riemannian metrics as comparable as possible. We then select pairs of points and we compute the distance between the curves that correspond to the pull-back and our proposed conformal metric. More details in App. E.3.

The results show that theoretical analysis in Sec. 4.2 is reasonable. In particular, for the *hole* case we see that both metrics behave similarly, since the paths avoid crossing the regions in $\mathcal{Z}$ with close to zero density (Prop. 1). This meaningful behavior is useful in practice as the shortest paths are pulled towards the latent codes for both metrics. However, some outliers still exist, as the geometry induced by our $\mathbf{M}_\psi(\cdot)$ is not exactly the same to the one of $\mathbf{M}_\theta(\cdot)$. This is apparent in the *ball* case, where the metrics have exactly the opposite behavior (Prop. 4). The ball samples increase the prior in $\mathcal{Z}$, while in the same region the uncertainty of $g(\cdot)$ increases as well. This causes the shortest paths to have a contrastive behavior. While in the *normal* case the two metrics result in similar curves (see Prop. 3). However, we show a pair of points where the path of $\mathbf{M}_\psi(\cdot)$ crosses a region with higher density and the path of $\mathbf{M}_\theta(\cdot)$ not, but the two curves are still similar. Therefore, if the data lie uniformly near a manifold in $\mathcal{X}$, we expect the metrics to behave similarly, due to the relation of the prior to the uncertainty of $g(\cdot)$.

5.2.1 Efficiency and robustness of the metrics

We investigate the behavior of the metrics as the dimension of $\mathcal{Z}$ increases, as well as the influence this has on the computation of shortest paths. We use the MNIST digits 0,1,2, which we project with PCA to 100 dimensions and we train a VAE for each $d = [2, 3, 5, 10]$ using our proposed prior. Also, we train post-hoc the RBF network to get the $\mathbf{M}_\theta(\cdot)$. Moreover, to make the metrics comparable we rescale them so that the maximum magnitude on the latent codes is equal to 1. For additional details see App. E.4.

We compute the magnitude on the latent codes and we see from the results in Fig. 6 that $\mathbf{M}_\psi(\cdot)$ is robust as dimensionality increases. This means that our prior is non-zero on the representations and relatively similar across them. In addition, we sample uniformly in the bounding box of the latent codes and we compute the magnitude, which shows that indeed our prior is non-zero only near the representations (see App. E.4). In contrast, $\mathbf{M}_\theta(\cdot)$ is not robust due to the RBF network. Due to the curse of dimensionality, the kernel output is nearly zero for some of the latent codes on the boundaries, which results in high magnitudes.

Additionally, we randomly sample 10 points per cluster and we compute the pairwise distances within each

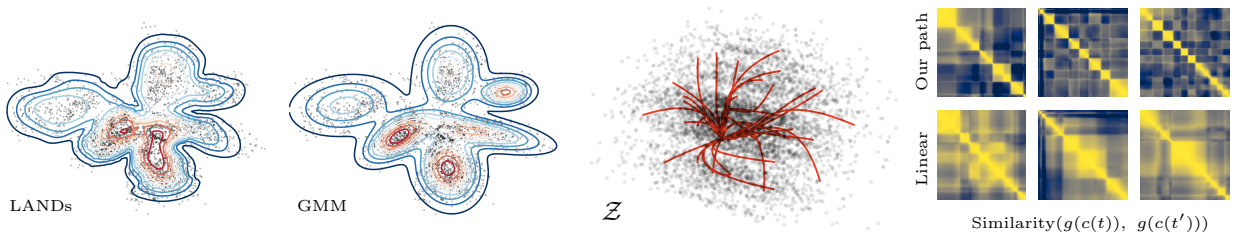


Figure 5: *Left*: The mixture models on cortex data. *Right*: The shortest paths in chemical compounds $\mathcal{Z}$ respect the structure of the latent representations. We also show the similarity of the generated samples along 3 interpolants. Our path generates relatively more explicit and diverse samples compared to the straight line.

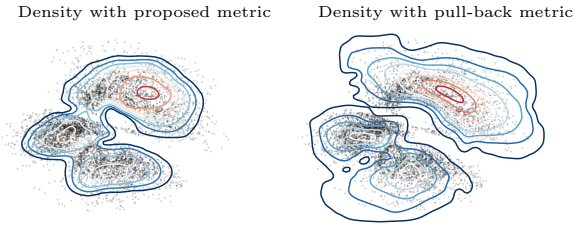


Figure 6: Due to the robustness of our proposed metric the mixture of LANDs density adapts better to the representations. In contrast, the pull-back metric is very high on some boundary points, so the curve lengths are overestimated which affects negatively the fitting of the mixture model.

cluster, in order to investigate the influence of the metrics on the shortest paths. The results in Fig. 6 show that $\mathbf{M}_\psi(\cdot)$ is highly efficient when computing shortest paths, even when d increases. The reason is that the corresponding ODEs system is simpler, more stable and also easier to solve. While $\mathbf{M}_\theta(\cdot)$ mainly due to the RBF, results in an unstable ODEs system, as well as, only evaluating the metric and its derivative is significantly more expensive. Consequently, the computation of the paths is very slow, while many times the solver fails ($> 25\%$). Further details in App. E.4.

5.3 Statistical models on manifolds

We fit a mixture of locally adaptive normal distributions (LANDs) defined on Riemannian manifolds with density $\rho(\mathbf{z}) = C(\boldsymbol{\mu}, \boldsymbol{\Gamma}) \cdot \exp(-0.5 \cdot \langle \mathrm{Log}_{\boldsymbol{\mu}}(\mathbf{z}), \boldsymbol{\Gamma} \cdot \mathrm{Log}_{\boldsymbol{\mu}}(\mathbf{z}) \rangle)$, mean $\boldsymbol{\mu} \in \mathbb{R}^d$, precision $\boldsymbol{\Gamma} \in \mathbb{R}^{d \times d}_{\succ 0}$ and normalization constant $C(\boldsymbol{\mu}, \boldsymbol{\Gamma})$ [Arvanitidis et al., 2016]. This is a flexible model but computationally expensive since it is fitted with gradient descent based on $\mathrm{Log}_{\boldsymbol{\mu}}(\cdot)$ and $\mathrm{Exp}_{\boldsymbol{\mu}}(\cdot)$ (see App. A). In Fig. 6 we show the result on the latent codes of the previous experiment. Due to the robustness of our $\mathbf{M}_\psi(\cdot)$ the density adapts better. In contrast, outliers with high $\mathbf{M}_\theta(\cdot)$ cause underestimated precisions. Also, the running times are respectively 10 min and 2 hours, because the ODEs system (Eq. 1) for $\mathbf{M}_\psi(\cdot)$ is significantly more efficient.

5.4 Applications in life sciences

We consider potential applications of the proposed metric in real world problems. Note that our setting is rather simplified and specialized models for such data exist. For the details see App. E.6.

We train a VAE on mouse cortex cell data that has a natural clustering [Zeisel et al., 2015]. In Fig. 5 we compare a mixture of LANDs with a Gaussian mixture model (GMM), where we see that the LANDs adapts better to the representations, which can be useful for exploratory data analysis by experts (see App. E.6 for individual components). Also, we can use the principal geodesics as a form of local *disentanglement*, as they represent the directions with highest variance on the data manifold (see App. E.6).

We train a recurrent VAE using chemical compounds from the ZINC database [Sterling and Irwin, 2015]. This type of data has an inherent natural structure that we can capture in $\mathcal{Z} = \mathbb{R}^3$ with our $\mathbf{M}_\psi(\cdot)$, even when the $g(\cdot)$ cannot induce the pull-back metric. We see in Fig. 5 that our paths explore and respect the learned nonlinear structure, as our interpolants generate diverse and explicit samples compared to the fuzzy ones of the straight line (see App. E.6). This amounts to interpretable and meaningful interpolations shown to reveal biological information [Detlefsen et al., 2020].

6 Conclusion

We propose to approximate the geometry of a data manifold in the latent space of a generative model using a Riemannian metric that is inversely proportional to a learnable prior. In addition, we propose a suitable energy-based model for the learnable prior in a VAE context. Our analysis shows that the proposed metric is a sensible approximation to the true pull-back metric, while being significantly more efficient and robust. Apart from its usefulness, our approach does not limit the generator's architecture. Also, it provides a way in future work to consider inductive biases based on interpretable geometric quantities of the metric.

References

Aneja, J., Schwing, A., Kautz, J., and Vahdat, A. (2020). NCP-VAE: Variational Autoencoders with Noise Contrastive Priors. In *arXiv preprint*.

Arvanitidis, G., Hansen, L. K., and Hauberg, S. (2016). A locally adaptive normal distribution. In *Neural Information Processing Systems (NeurIPS)*.

Arvanitidis, G., Hansen, L. K., and Hauberg, S. (2017). Maximum likelihood estimation of riemannian metrics from euclidean data. In *Geometric Science of Information (GSI)*.

Arvanitidis, G., Hansen, L. K., and Hauberg, S. (2018). Latent space oddity: on the curvature of deep generative models. In *International Conference on Learning Representations (ICLR)*.

Arvanitidis, G., Hauberg, S., Hennig, P., and Schober, M. (2019). Fast and robust shortest paths on manifolds learned from data. In *Artificial Intelligence and Statistics (AISTATS)*.

Arvanitidis, G., Hauberg, S., and Schölkopf, B. (2020). Geometrically Enriched Latent Spaces. In *arXiv preprint*.

Bauer, M. and Mnih, A. (2019). Resampled priors for variational autoencoders. In *Artificial Intelligence and Statistics (AISTATS)*.

Dai, B. and Wipf, D. (2019). Diagnosing and enhancing VAE models. In *International Conference on Learning Representations*.

Detlefsen, N. S., Hauberg, S., and Boomsma, W. (2020). What is a meaningful representation of protein sequences? In *arXiv preprint*.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. (2016). Density estimation using real NVP. In *arXiv preprint*.

do Carmo, M. (1992). *Riemannian Geometry*. Mathematics (Boston, Mass.). Birkhäuser.

Eklund, D. and Hauberg, S. (2019). Expected path length on random manifolds. In *arXiv preprint*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Neural Information Processing Systems (NeurIPS)*.

Hauberg, S. (2018). Only Bayes should learn a manifold. In *arXiv preprint*.

Hauberg, S., Freifeld, O., and Black, M. (2012). A Geometric Take on Metric Learning. In *Neural Information Processing Systems (NeurIPS)*.

Hennig, P. and Hauberg, S. (2014). Probabilistic solutions to differential equations and their application to riemannian statistics. In *Artificial Intelligence and Statistics (AISTATS)*.

Kingma, D. P. and Welling, M. (2014). Auto-Encoding Variational Bayes. In *International Conference on Learning Representations (ICLR)*.

Lawrence, N. (2005). Probabilistic Non-linear Principal Component Analysis with Gaussian Process Latent Variable Models. *J. Mach. Learn. Res.*

Lebanon, G. (2002). Learning riemannian metrics. In *Uncertainty in Artificial Intelligence (UAI)*.

LeCun, Y., Chopra, S., Hadsell, R., Huang, F. J., and et al. (2006). A tutorial on energy-based learning. In *Predicting Structured Data*. MIT Press.

Lee, J. (2018). *Introduction to Riemannian Manifolds*. Springer.

Mohamed, S., Rosca, M., Figurnov, M., and Mnih, A. (2020). Monte carlo gradient estimation in machine learning. *Journal of Machine Learning Research*.

Pang, B., Han, T., Nijkamp, E., Zhu, S.-C., and Wu, Y. N. (2020). Learning latent space energy-based prior model. In *arXiv preprint*.

Pennec, X. (2006). Intrinsic Statistics on Riemannian Manifolds: Basic Tools for Geometric Measurements. *Journal of Mathematical Imaging and Vision*.

Pfau, D., Higgins, I., Botev, A., and Racanière, S. (2020). Disentangling by Subspace Diffusion. In *Neural Information Processing Systems (NeurIPS)*.

Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning (ICML)*.

Sterling, T. and Irwin, J. J. (2015). ZINC 15 – Ligand Discovery for Everyone. *Journal of Chemical Information and Modeling*.

Tenenbaum, J. B., Silva, V. d., and Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*.

Tomczak, J. M. and Welling, M. (2018). VAE with a VampPrior. In *Artificial Intelligence and Statistics (AISTATS)*.

Tosi, A., Hauberg, S., Vellido, A., and Lawrence, N. D. (2014). Metrics for Probabilistic Geometries. In *Uncertainty in Artificial Intelligence (UAI)*.

Yang, T., Arvanitidis, G., Fu, D., Li, X., and Hauberg, S. (2018). Geodesic clustering in deep generative models. In *arXiv preprint*.

Zeisel, A., Muñoz-Manchado, A. B., Codeluppi, S., Lönnerberg, P., La Manno, G., Juréus, A., Marques, S., Munguba, H., He, L., Betsholtz, C., Rolny, C., Castelo-Branco, G., Hjerling-Leffler, J., and Linnarsson, S. (2015). Cell types in the mouse cortex and hippocampus revealed by single-cell rna-seq. *Science*.

Supplementary Material:
A prior-based approximate latent Riemannian metric

Negative social impact. In this work we provide a methodology that enables us to use the latent variables of a deep generative model in a geometrically meaningful way, by computing shortest paths that respect the intrinsic structure of the data manifold. As we showed in our experiments, this could be potentially useful for applications in life sciences and we cannot see our work having an immediate negative social impact. Nevertheless, generative models have been also used for malicious purposes e.g. generating fake content. However, we believe that the merits outweigh the possible risks.

A Riemannian geometry

A Riemannian manifold $\mathcal{M}$ is a smooth manifold together with a Riemannian metric that defines a smoothly changing local inner product acting on the tangent space [Lee, 2018; do Carmo, 1992]. The most intuitive way to conceptualize a Riemannian manifold is as a d-dimensional hypersurface embedded in a higher-dimensional ambient space $\mathcal{X} = \mathbb{R}^D$. The simplest Riemannian metric in this case is the restriction of the Euclidean metric $\mathbb{I}_D$ on each tangent space $\mathcal{T}_{\mathbf{x}}\mathcal{M}$, where the tangent space is a d-dimensional vector space that touches $\mathcal{M}$ tangentially at $\mathbf{x} \in \mathcal{M}$. So a tangent vector $\mathbf{v} \in \mathcal{T}_{\mathbf{x}}\mathcal{M}$ is actually a vector in $\mathbb{R}^D$. By definition, a smooth manifold can be covered by a collection of charts. A chart can be seen as a parametrization of a neighborhood on $\mathcal{M}$ formally written as $\phi_j : \mathcal{U}_j \subset \mathcal{M} \to \mathcal{H}_j \subset \mathbb{R}^d$. In other words, a chart gives us d-dimensional coordinates that represent the points in a neighborhood $\mathcal{U}_j$. Moreover, on a smooth manifold the charts are diffeomorphisms by definition. However, for simplicity, we assume that a global chart $h(\cdot)$ exists, which gives a global paramtrization of the manifold, so we can write that $h : \mathcal{H} \to \mathcal{M}$. The space $\mathcal{H}$ is known as the *intrinsic coordinates*.

Since $h(\cdot)$ is a differomorphism, we know that $\mathbf{J}_h : \mathcal{H} \to \mathbb{R}^{D\times d}$ is full-rank, and hence, we can uniquely map a vector $\overline{\mathbf{v}} \in \mathcal{H}$ from the intrinsic coordinates to a tangent vector $\mathbf{v} \in \mathcal{T}_{\mathbf{x}}\mathcal{M}$ as $\mathbf{v} = \mathbf{J}_h(\mathbf{z})\overline{\mathbf{v}}$. Assuming that the Riemannian metric in the ambient space $\mathcal{X}$ is the Euclidean $\mathbf{M}_{\mathcal{X}}(\cdot) = \mathbb{I}_D$, we can define the inner product in each tangent space as $\langle \mathbf{v}, \mathbf{v} \rangle_{\mathbf{x}} = \langle \overline{\mathbf{v}}, \mathbf{M}_{\mathcal{H}}(\mathbf{z})\overline{\mathbf{v}} \rangle$. Therefore, on the tangent space, which is actually a d-dimensional vector space the metric now is $\mathbf{M}_{\mathcal{H}}(\mathbf{z}) = \mathbf{J}_h(\mathbf{z})^\intercal \mathbf{J}_h(\mathbf{z})$ which changes for each point $\mathbf{x} = h(\mathbf{z})$. Intuitively, on the tangent space we represent a "linearized" view of $\mathcal{M}$ with respect to a base point $\mathbf{x} \in \mathcal{M}$. When working directly in the embedding space, the linear representation $\mathbf{v}$ is scaled by the metric $\mathbf{M}_{\mathcal{X}}(\mathbf{x})$, and equivalently, the $\overline{\mathbf{v}}$ is scaled by $\mathbf{M}_{\mathcal{H}}(\mathbf{z})$ when working in the intrinsic coordinates.

In addition, the metric $\mathbf{M}_{\mathcal{H}}(\cdot)$ appears in the intrinsic coordinates $\mathcal{H}$ and represents the amount of distortion caused to the infinitesimal volume element $d\mathbf{z}$ when mapped through $h(\cdot)$ on $\mathcal{M}$. Also, due to the fact that the chart is a diffeormorphism, we get that the metric is smooth as it is based on the Jacobian of $h(\cdot)$. In fact, the embedding of a smooth manifold $\mathcal{M}$ in a higher dimensional ambient space with $\mathbf{M}_{\mathcal{X}}(\cdot)$ directly induces a Riemannian metric in the intrinsic coordinates $\mathcal{H}$. In this work we assume that $\mathbf{M}_{\mathcal{X}}(\cdot) = \mathbb{I}_D$.

This Riemannian metric allows us to compute distances between points on $\mathcal{M}$. Actually, it represents the distortions of the infinitesimal distance and volume element. Let a curve $\gamma : [0, 1] \to \mathcal{M} \subset \mathcal{X}$ with $\gamma(0) = \mathbf{x}$ and $\gamma(1) = \mathbf{y}$. We can measure the curve length on $\mathcal{M}$ by considering the curve simply lying in $\mathcal{X}$, so we get

$$\ell[\gamma] = \int_0^1 \sqrt{\langle \dot{\gamma}(t), \dot{\gamma}(t) \rangle_{\gamma(t)}} dt = \int_0^1 \sqrt{\langle \dot{\gamma}(t), \mathbb{I}_D \dot{\gamma}(t) \rangle} dt = \int_0^1 \sqrt{\langle \dot{c}(t), \mathbf{M}_{\mathcal{H}}(c(t))\dot{c}(t) \rangle} dt = \ell[c], \tag{7}$$

where $\dot{\gamma}(t) = \partial_t \gamma(t) \in \mathcal{T}_{\gamma(t)}\mathcal{M}$ is the velocity of the curve and $\gamma(t) = h(c(t))$. Here, we assumed that the metric of $\mathcal{X}$ is the Euclidean, however, other meaningful Riemannian metrics could have been used [Arvanitidis et al., 2020]. This result shows that instead of computing the length of a curve on $\mathcal{M} \subset \mathcal{X}$ we can equivalently compute it in the intrinsic coordinates $\mathcal{H}$.

Moreover, we can find the shortest path i.e. the curve with minimum length, by optimizing the functional Eq. 7. However, it is known that the length is parametrization invariant. In other words, we can reparametrize t and

get still the same length. Instead, the energy is not invariant under reparametrizations of t, and thus, we can find the curve with minimum energy by optimizing the energy functional either in $\mathcal{X}$ or in $\mathcal{H}$ as

$$\gamma^* = \underset{\gamma}{\operatorname{argmin}} \int_0^1 \langle \dot{\gamma}(t), \dot{\gamma}(t) \rangle_{\gamma(t)} dt \qquad \equiv \qquad c^* = \underset{c}{\operatorname{argmin}} \int_0^1 \sqrt{\langle \dot{c}(t), \mathbf{M}_{\mathcal{H}}(c(t)) \dot{c}(t) \rangle} dt. \tag{8}$$

In $\mathcal{H}$ we can apply the Euler-Lagrange equations which gives us a system of second order nonlinear ordinary differential equations [Arvanitidis et al., 2018]

$$\ddot{c}(t) = -\frac{1}{2} \mathbf{M}_{\mathcal{H}}^{-1}(c(t)) \Big[2 \cdot \partial_{c(t)} \mathbf{M}_{\mathcal{H}}(c(t)) - \partial_{c(t)} \text{vec}[\mathbf{M}_{\mathcal{H}}(c(t))]^{\intercal} \Big] (\dot{c}(t) \otimes \dot{c}(t)), \tag{9}$$

that we need to solve in order to find the curve that minimizes the energy. The resulting curve is a minimizer of the length as well. Here, $\otimes$ is the Kronocker product, vec$[\cdot]$ stacks the columns of the matrix and the term $\partial_{c(t)} \mathbf{M}_{\mathcal{H}}(c(t)) = [\partial_{c_1(t)} \mathbf{M}_{\mathcal{H}}(c(t)), \dots, \partial_{c_d(t)} \mathbf{M}_{\mathcal{H}}(c(t))] \in \mathbb{R}^{d \times d^2}$ where $\partial_{c_j(t)} \mathbf{M}_{\mathcal{H}}(c(t)) \in \mathbb{R}^{d \times d}$ is the partial derivative of the metric $\mathbf{M}_{\mathcal{H}}(c(t))$ with respect to the j-th component of the curve.

We can find the shortest path by solving the ODE system above as a boundary value problem (BVP) with $c(0) = \mathbf{z_x}$ and $c(1) = \mathbf{z_y}$ the corresponding points in $\mathcal{H}$ of $\gamma(0) = \mathbf{x}$ and $\gamma(1) = \mathbf{y}$. Unfortunately, for general Riemannian manifolds the analytic solution is intractable, and thus, we rely on approximate numerical solutions [Arvanitidis et al., 2019; Hennig and Hauberg, 2014; Yang et al., 2018].

In order to perform computations on $\mathcal{M}$ or equivalently in the intrinsic coordinates $\mathcal{H}$ we use two operators. The logarithmic map $\text{Log}_{\mathbf{x}}(\mathbf{y}) = \mathbf{v} \in \mathcal{T}_{\mathbf{x}}\mathcal{M}$ takes two points $\mathbf{x}, \mathbf{y} \in \mathcal{M}$ and returns a tangent vector on the tangent space of $\mathbf{x}$. The vector $\mathbf{v}$ can be seen as the initial velocity of the curve that starts at $\mathbf{x}$ and on time $t = 1$ reaches the point $\mathbf{y}$. Since $\mathcal{T}_{\mathbf{x}}\mathcal{M}$ is a vector space, this operator provides a linear representation of (a neighborhood on) $\mathcal{M}$ with respect to the base point $\mathbf{x} \in \mathcal{M}$. In practice, we compute the logarithmic map in the intrinsic coordinates by solving the ODE system as a Boundary Value Problem (BVP). The inverse operator is the exponential map that takes a point $\mathbf{x} \in \mathcal{M}$ and a vector $\mathbf{v} \in \mathcal{T}_{\mathbf{x}}\mathcal{M}$ and returns a geodesic $\text{Exp}_{\mathbf{x}}(t \cdot \mathbf{v}) = \gamma(t)$ with $\gamma(1) = \mathbf{y}$. Again, we implement this operator in the intrinsic coordinates $\mathcal{H}$ by solving the ODE system as an Initial Value Problem (IVP). The length of a tangent vector, as it lies on a tangent space, it is computed under the Riemannian metric and it is by definition length$[\gamma] = \langle \mathbf{v}, \mathbf{v} \rangle_{\mathbf{x}} = \langle \overline{\mathbf{v}}, \mathbf{M}_{\mathcal{H}}(\mathbf{z}) \overline{\mathbf{v}} \rangle = \text{length}[c]$, where $\gamma(t)$ and $c(t)$ the geodesics on $\mathcal{M}$ and $\mathcal{H}$ respectively. We can rescale or reparametrize the intrinsic vector $\overline{\mathbf{v}}$ to $\widetilde{\mathbf{v}}$ such that the metric only locally to become Euclidean $\widetilde{\mathbf{M}}_{\mathcal{H}}(\cdot) = \mathbb{I}_d$ so the length$[c] = \langle \widetilde{\mathbf{v}}, \widetilde{\mathbf{v}} \rangle$. The new representation $\widetilde{\mathbf{v}}$ is known as the *normal coordinates*.

For clarification, the tangent vector $\mathbf{v} \in \mathcal{T}_{\mathbf{x}}\mathcal{M}$ in the ambient space $\mathcal{X}$ is a vector in $\mathbb{R}^D$ that is tangential to a d-dimensional $\mathcal{M}$ at the point $\mathbf{x} \in \mathcal{M}$, as the tangent space is a hyperplane that touches tangentially $\mathcal{M}$ at the point $\mathbf{x}$. Hence, we can get a linear representation of $\mathcal{M}$ on each tangent space, which is a d-dimensional vector space. On the other hand, an example of intrinsic coordinates for $\mathcal{M}$ can be see in Fig. 2. In the intrinsic coordinates $\mathcal{H} \subseteq \mathbb{R}^d$ the tangent space at a point $\mathbf{z} \in \mathcal{H}$ is simply the $\mathbb{R}^d$ centered at $\mathbf{z}$. So we can linearly represent the intrinsic coordinates with respect to a base point $\mathbf{z}$ as vectors $\overline{\mathbf{v}} \in \mathbb{R}^d$ centered at $\mathbf{z}$. Actually, $\overline{\mathbf{v}} \in \mathbb{R}^d$ corresponds to the intrinsic representation of the vector $\mathbf{v} \in \mathbb{R}^D$ on the d-dimensional vector space $\mathcal{T}_{\mathbf{x}}\mathcal{M}$.

The analysis above shows that essentially the Riemannian metric $\mathbf{M}_{\mathcal{H}}(\cdot)$ and the intrinsic coordinates $\mathcal{H}$ are enough in order to compute shortest paths on a manifold $\mathcal{M}$. This further implies that as long as these quantities are given, then $\mathcal{M}$ could even be an abstract manifold. Unfortunately, in the setting where the manifold is implied by data that lie in $\mathcal{X}$, the Riemannian metric is usually unknown. Moreover, a unique chart rarely exists. In this case, we use a trick to capture the geometry of the data manifold.

Specifically, let $\mathcal{Z} \subseteq \mathbb{R}^{d'}$ and we learn an at least twice differentiable function $g : \mathcal{Z} \to \mathcal{M} \subset \mathcal{X}$ but not necessarily a differomorphism. Then, following the previous analysis we can induce a Riemannian metric $\mathbf{M}_{\mathcal{Z}} : \mathcal{Z} \to \mathbb{R}^{d' \times d'}_{\succ 0}$. The high level idea is that if a global chart existed and $\mathcal{Z} = \mathcal{H}$ with $g(\cdot) \approx h(\cdot)$ then the $\mathbf{M}_{\mathcal{Z}}(\cdot) \approx \mathbf{M}_{\mathcal{H}}(\cdot)$. Even if this is rarely the case, the $\mathbf{M}_{\mathcal{Z}}(\cdot)$ is still able to capture the geometry of the neighborhoods on $\mathcal{M}$ which approximates e.g. a submanifold on $\mathcal{M}$ if $d' < d$. This metric is known as the pull-back metric. However, in the data manifold regime as it has been shown from previous works [Arvanitidis et al., 2018; Tosi et al., 2014; Hauberg, 2018; Eklund and Hauberg, 2019] the $g(\cdot)$ should be a stochastic generator in order to capture properly the geometry of $\mathcal{M}$ in a latent space $\mathcal{Z}$.

Let $\mathcal{Z} = \mathbb{R}^{d'}$, which is a smooth manifold with a trivial tangent space, and consider a Riemannian metric $\mathbf{M}_{\mathcal{Z}}(\cdot)$ therein. Computing curve lengths under this metric transforms $\mathcal{Z}$ into a Riemannian manifold. In some sense,

this Riemannian manifold "imitates" or "captures approximately" the geometry of $\mathcal{M}$. In practice, the metric scales the distances locally in $\mathcal{Z}$, so it changes the way we measure curve lengths therein.

The proposed conformal metric in this paper is one way to approximate the behavior of the computationally expensive $\mathbf{M}_{\mathcal{Z}}(\cdot)$, since evaluating and derivating this metric relies on expensive computations. As we showed in the main paper (see Sec. 4), the new metric in many cases is a sensible approximation to the actual pull-back metric. We analyzed theoretically the behavior of the two metrics, and we argued that due to their actual definition both metrics induce the same "topological" structure in $\mathcal{Z}$. In other words, in both cases the shortest paths are pulled towards the training latent codes. Of course, there are also problematic cases where the two metrics have the exact opposite behavior. However, one advantage of the proposed metric is the ability to take it into account during training. Therefore, we can add regularizers to make the two metrics more similar or even to include interepretable inductive biases in the form of geometric regularizers.

Identifiability in our context considers the preservation of the distance measure between points under diffeomorphic reparametrizations of the intrinsic coordinates. In particular, let two functions $g_1 : \mathcal{Z}_1 \subseteq \mathbb{R}^d \to \mathcal{M} \subset \mathcal{X}$ and $g_2 : \mathcal{Z}_2 \subseteq \mathbb{R}^d \to \mathcal{M} \subset \mathcal{X}$, where $g_2(\cdot) = T \circ g_1(\cdot)$ with $T(\cdot)$ a diffeomorphic transformation. The reparametrization directly implies that for any pair of points $\mathbf{x}_1,\ \mathbf{y}_1 \in \mathcal{Z}_1$ and the corresponding points $\mathbf{x}_2,\ \mathbf{y}_2 \in \mathcal{Z}_2$ the Euclidean distance in general is $||\mathbf{x}_1 - \mathbf{y}_1||_2 \neq ||\mathbf{x}_2 - \mathbf{y}_2||_2$. However, the curve length on the manifold $\mathcal{M}$ between $\mathbf{x} = g_1(\mathbf{x}_1) = g_2(\mathbf{x}_2)$ and $\mathbf{y} = g_1(\mathbf{y}_1) = g_2(\mathbf{y}_2)$ does not change. Note that when we measure the length of a curve using the pull-back metric in $\mathcal{Z}_1$ or $\mathcal{Z}_2$, then we actually measure the length directly on $\mathcal{M}$. Therefore, if both functions $g_1(\cdot)$ and $g_2(\cdot)$ generate $\mathcal{M}$, then we know that the curve length is the same in both parametrizations when measured under each corresponding pull-back metric. In other words, if for any arbitrary learned parametrization $g_j(\cdot)$ the generated $\mathcal{M}$ remains the same, then the distance measured under the corresponding pull-back is invariant. Note that this is also the case when the distribution remains invariant after the transformation $T(\cdot)$ as $\mathbf{z}_1 \sim p(\mathbf{z})$ and $T(\mathbf{z}_1) = \mathbf{z}_2 \sim p(\mathbf{z})$.

B Riemannian metrics from data

There are several ways to construct a Riemannian metric from a given set of observations. Here, we present some methods that have been proposed previously in the literature.

Hauberg et al. [2012] proposed a Riemannian metric as a weighted sum of a predefined set of metric tensors. In particular, let $\mathbf{M}_{1:K} \in \mathbb{R}^{D\times D}_{\succ 0}$ a predefined set of positive definite metric tensors centered at points $\mathbf{x}_{1:K} \in \mathbb{R}^D$. Then, the metric at new points $\mathbf{x}$ is computed as

$$\mathbf{M}(\mathbf{x}) = \sum_{k=1}^{K} \widetilde{\mathbf{w}}_k(\mathbf{x})\mathbf{M}_k, \tag{10}$$

where $\mathbf{w}_k(\mathbf{x}) = \exp\left(-\frac{||\mathbf{x}_k - \mathbf{x}||_2^2}{2\sigma^2}\right)$, $\sigma > 0$ the bandwidth or support of the kernel and $\widetilde{\mathbf{w}}_k(\mathbf{x}) = \frac{\mathbf{w}_k(\mathbf{x})}{\sum_{l=1}^{K} \mathbf{w}_k(\mathbf{x})}$. In this case, the predefined metrics can be estimated using additional information e.g. labels. The bandwidth controls how large is the neighborhood from which we consider the predefined metrics and it is rather hard to find the optimal σ. Especially, when the dimension of the space is high, where the curse of dimensionality occurs. Finally, one downside of this metric is that as we move away from the training data, the magnitude does not necessarily increase. Because the normalized weights still select some of the predefined metrics.

In a similar spirit Arvanitidis et al. [2016] proposed an unsupervised approach to construct a Riemannian metric from data. In particular, the metric is defined as the inverse local diagonal covariance matrix, so the diagonal elements of the metric are computed as

$$M_{jj}(\mathbf{x}) = \left[\sum_{n=1}^{N} \mathbf{w}_n(\mathbf{x})(x_{nj} - x_j) + \rho\right]^{-1}, \tag{11}$$

where $\mathbf{w}_n(\mathbf{x}) = \exp\left(-\frac{||\mathbf{x}_n - \mathbf{x}||_2^2}{2\sigma^2}\right)$, the parameter $\sigma > 0$ is again the bandwidth and $\rho > 0$ a parameter to upper bound the metric. The parameter σ in some sense controls the curvature of the metric i.e., how fast the metric changes. However, again finding the optimal parameter is a challenging task. Regarding ρ, it is chosen as a small value such that to pull shortest paths near the data.

A simple conformally flat Riemannian metric has been proposed by Arvanitidis et al. [2020], defined as

$$\mathbf{M}(\mathbf{x}) = (\alpha \cdot r(\mathbf{x}) + \beta)^{-1} \cdot \mathbb{I}_D, \tag{12}$$

where $\alpha, \beta > 0$ are parameters to lower and upper bound the metric. Here, the function $r(\cdot)$ is modeled as a positive RBF $r(\mathbf{x}) = \mathbf{w}^\intercal \phi(\mathbf{x})$, with $\mathbf{w} \in \mathbb{R}^K_{>0}$ and $\phi_k(\mathbf{x}) = \exp\left(-\frac{||\mathbf{c}_k - \mathbf{x}||_2^2}{2\sigma^2}\right)$ for some centers $\mathbf{c}_k$ near the training data, such that $r(\mathbf{x}) \to 1$ near the training data and $r(\mathbf{x}) \to 0$ as we move away from them. Again here, the problem is how to find the optimal parameters σ as well as the kernel behavior in higher dimensions.

Even if the kernel based Riemannian metrics above are simple and meaningful, their performance is rather limited. The main problem is the selection of the bandwidth σ, as well as the behavior of the kernel especially in higher dimensions. For this reason, another line of work proposed to learn Riemannian metrics in the latent space of a generative model. This approach allows to reduce the dimensionality of the problem. Even if theoretically the resulting metrics capture precisely the data manifold's geometry, in practice, they rely on some form of a kernel as well. Moreover, their usability is hindered by inevitable computational complexity.

Here we focus on deep generative models and more specifically on Variational Auto-Encoders [Kingma and Welling, 2014; Rezende et al., 2014]. However, the same analysis can be done in the context of Gaussian Processes with GPLVMs [Tosi et al., 2014]. Let a stochastic generator $\mathbf{x} = g(\mathbf{z}) = \mu_\theta(\mathbf{z}) + \text{diag}(\sigma_\theta(\mathbf{z})) \cdot \varepsilon$ with $\varepsilon \sim \mathcal{N}(0, \mathbb{I}_D)$. Obviously, this stochastic function is not differentiable as it is a non-smooth function due to ε. Instead, fixing ε makes the function smooth. This can be seen as generating a surface $g(\mathcal{Z}) \subset \mathcal{X}$ for a fixed noise vector ε, which in expectation results in a distribution of points that converges to the actual generative process. Eklund and Hauberg [2019] viewed this step as a random projection of a smooth surface that lies in a higher dimensional space $[\mu_\theta(\mathbf{z}), \sigma_\theta(\mathbf{z})] \in \mathbb{R}^{D^2}$ using the projection matrix block_diag$([\mathbb{I}_D, \text{diag}(\varepsilon)]) \in \mathbb{R}^{D \times D^2}$.

Therefore, we can compute the Jacobian of $g(\cdot)$ and the expected Riemannian metric in $\mathcal{Z}$ as

$$\mathbf{M}_\theta(\mathbf{z}) = \mathbb{E}_\varepsilon[\mathbf{J}_g^\intercal(\mathbf{z})\mathbf{J}_g(\mathbf{z})] = \mathbf{J}_{\mu_\theta}^\intercal(\mathbf{z})\mathbf{J}_{\mu_\theta}(\mathbf{z}) + \mathbf{J}_{\sigma_\theta}^\intercal(\mathbf{z})\mathbf{J}_{\sigma_\theta}(\mathbf{z}), \tag{13}$$

which is known as the pull-back metric. This an interpretable and meaningful metric, since the second term that is based on the uncertainty makes sure that the shortest paths prefer to stay in regions of the latent space with low uncertainty. To achieve this we need the $\sigma_\theta(\cdot)$ to increase as we move further from the latent codes. The solution proposed by Arvanitidis et al. [2018] is to utilize a positive RBF network to model the precision i.e. the inverse variance. So, as we move further from the latent codes due to the RBF behavior the uncertainty increases. Even if this Riemannian metric seems as a reasonable solution, as we discussed in the main paper (see Sec. 4) it comes with some practical disadvantages.

For this metric, we also need to set a hyperparameter for the variance function $\sigma_\theta^2(\mathbf{z}) = \xi_\theta(\mathbf{z})^{-1}$ where

$$\xi_\theta(\mathbf{z}) = \mathbf{W}\ \boldsymbol{\kappa}(\mathbf{z}) + \zeta \tag{14}$$

with $\mathbf{W} \in \mathbb{R}^{D \times K}_{>0}$, $\kappa_k(\mathbf{z}) = \exp(-0.5 \cdot \lambda \cdot ||\mathbf{z} - \mathbf{z}_k||_2^2)$ and $\zeta > 0$ is the lower bound for the precision or equivalently an upper bound for the uncertainty. Hence, ζ implicitly influences the metric, as it controls the point where $\mathbf{J}_{\sigma_\theta}(\cdot)$ becomes close to constant. Practically, it does not allow the precision to become zero. Before $\mathbf{J}_{\sigma_\theta}(\cdot)$ becoming close to constant, the corresponding part of the pull-back metric achieves its maximum value. This sets the boundaries around the latent codes, which represents in some sense the topology of the data manifold in $\mathcal{Z}$. Also, it affects the maximum magnitude.

As we discussed in the main paper, usually the training of the VAE is done using a deep neural network $\overline{\sigma}_\theta^2(\cdot)$ to model the uncertainty of the generator. We use the same modeling choice during the training. Then, we train post-hoc the RBF network, as a regression problem or using again the ELBO while fixing the other functions. So a practical way to set ζ is after the first phase of the training to compute the mean variance $\overline{\sigma}^2_{\text{mean}} = \frac{1}{N \cdot D} \sum_{n=1}^{N} \sum_{j=1}^{D} [\overline{\sigma}_\theta^2(\mathbf{z}_n)]_j$ of the training latent codes. Then we can set $\zeta = (\alpha \cdot \overline{\sigma}^2_{\text{mean}})^{-1}$ where $\alpha > 0$ is a multiplicative factor e.g. $\alpha = 1000$. This is heuristic way to fix the hyperparameter ζ.

Rescaling the metrics. In order the magnitude to be as comparable as possible across different Riemannian metrics, we propose to scale each metric. As regards the pull-back metric we compute the magnitude on the training latent codes and we find the maximum $m_{\max} = \max_{\mathbf{z}_n} \sqrt{|\mathbf{M}_\theta(\mathbf{z})|}$. Then, we rescale the metric as

$$\mathbf{M}_\theta(\mathbf{z}) \triangleq \frac{1}{m_{\max}^{2/d}} \left[\mathbf{J}_{\mu_\theta}^\intercal(\mathbf{z})\mathbf{J}_{\mu_\theta}(\mathbf{z}) + \mathbf{J}_{\sigma_\theta}^\intercal(\mathbf{z})\mathbf{J}_{\sigma_\theta}(\mathbf{z})\right], \tag{15}$$

which ensures that the maximum magnitude on the training latent codes is 1. Here, if $m_{\max}$ is a huge value, then the metric is scaled dramatically especially on the rest of the latent codes. This is precisely what we see in Sec. 5 when we compare the robustness of the metrics.

Given a probability density $\nu_\psi(\cdot)$ function that represents the prior of the training latent codes, we proposed in this paper the conformally flat Riemannian metric

$$\mathbf{M}_\psi(\mathbf{z}) = m(\mathbf{z}) \cdot \mathbb{I}_d = (\alpha \cdot \nu_\psi(\mathbf{z}) + \beta)^{-2/d} \cdot \mathbb{I}_d, \tag{16}$$

where $\alpha,\ \beta > 0$ two hyperparameter that lower and upper bound the metric respectively. Interestingly, this metric does not depend in any way on a kernel. Also, the parameters ψ of the metric can be learned during the VAE training. As regards the hyperparameters, we can set $\beta = {}^1/_{m_{\max}}$ where $m_{\max}$ is the highest value that $\sqrt{|\mathbf{M}_\psi(\cdot)|}$ can get. Typically, the $m_{\max}$ is set to a large value like 100. Then, we find the lowest prior value on the training latent codes $\nu_{\min} = \min_{\mathbf{z}_n} \nu_\psi(\mathbf{z})$ and set the $\alpha = {}^{(1-\beta)}/_{\nu_{\min}}$. This ensures that the maximum magnitude on the training latent codes is 1. Since this number is $1 \ll m_{\max}$ the shortest paths will be pulled towards the training latent codes.

The proposed metric does not depend on a kernel, which makes it robust. Also, does not rely on any Jacobian computation, so it is efficient. While its hyperparameters can be fixed relatively easier, which makes it simple.

C Training details for the proposed prior

Our learnable prior is based on energy-based models [LeCun et al., 2006], and is defined as

$$\nu_\psi(\mathbf{z}) = \frac{\exp(f_\psi(\mathbf{z})) \cdot p(\mathbf{z})}{\mathcal{C}}, \tag{17}$$

where $f_\psi : \mathcal{Z} \to \mathbb{R}$ is a deep neural network and the base prior $p(\mathbf{z}) = \mathcal{N}(0, \mathbb{I}_d)$. Since $\mathcal{Z}$ is typically a low dimensional space, the normalization constant can be computed using naive Monte Carlo as

$$\mathcal{C} = \int_{\mathcal{Z}} \exp(f_\psi(\mathbf{z})) \cdot p(\mathbf{z}) d\mathbf{z} \approx \frac{1}{S} \sum_{s=1}^{S} \exp(f_\psi(\mathbf{z}_s)), \tag{18}$$

where $\mathbf{z}_s \sim p(\mathbf{z})$. In addition, when we train the VAE we include the latent codes of the training batch in the estimation of the normalization constant. This helps to prevent the function $f_\psi(\cdot)$ of getting extreme values. In theory, the samples from $p(\mathbf{z})$ should be enough such that to regularize the function $f_\psi(\cdot)$. However, especially in higher dimensions the number of samples S might not be large enough, in order to successfully regularize $f_\psi(\cdot)$. For this reason, we included the latent codes of the training batch, which we empirically observed to work well i.e., extreme values of $f_\psi(\cdot)$ and $\mathcal{C}$ do not occur. Of course, in the training objective of the VAE (see Eq. 4), due to the $\log(\cdot)$ that is applied on the constant, we used the log-sum-exp trick in order to stabilize the training.

Another training trick (which we did not use) is to regularize the prior using temperature. In particular, we can use a temperature parameter T in the exponent $\exp(T \cdot f_\psi(\mathbf{z}))$, for which a large T gives a more complex prior and a smaller T gives a smoother prior [LeCun et al., 2006]. Similarly, we can regularize the prior implicitly by applying standard regularization techniques for the parameters of $f_\psi(\cdot)$ e.g. L_2 regularization for the weights.

The actual contribution of the normalization constant is to regularize implicitly the prior to be close to zero in regions of $\mathcal{Z}$ with no latent codes. This is important, because it ensures that the magnitude increases as we move further from the latent codes. In other words, this helps to approximate well the structure (or topology) of the data manifold. However, after the training, the normalization constant does not affect anymore the metric, especially if we set the parameters α, β as explained above.

D Theoretical analysis of the proposed metric

We provide the details for the theoretical analysis in Sec. 4.2. Apart from the corresponding demonstrations in the main paper (see Sec. 5), we provide additional empirical evaluations of these results in App. E.3.

Proposition 1. *Let a learned $g(\cdot)$ with $\sigma^2_\theta(\cdot)$ an inverse RBF network and $\nu_\psi(\mathbf{z})$ the proposed prior. Then, the magnitude of the metrics $\mathbf{M}_\theta(\cdot)$ and $\mathbf{M}_\psi(\cdot)$ is maximum in the same region of $\mathcal{Z}$ where $\nu_\psi(\mathbf{z}) \to 0$.*

Proof. We consider first our proposed metric that is defined in Eq. 4.1 as $\mathbf{M}_\psi(\mathbf{z}) = (\nu_\psi(\mathbf{z}) + \beta)^{-2/d} \cdot \mathbb{I}_d$, where we use for simplicity $\alpha = 1$ and $\beta = 0$. So the magnitude is equal to $\sqrt{|\mathbf{M}_\psi(\mathbf{z})|} = \nu_\psi(\mathbf{z})^{-1}$, which means that the $\sqrt{|\mathbf{M}_\psi(\mathbf{z})|} \to +\infty$ when $\nu_\psi(\mathbf{z}) \to 0$.

As regards the pullback metric, this is defined in Eq. 5 as $\mathbf{M}_\theta(\mathbf{z}) = \mathbf{M}_{\mu_\theta}(\mathbf{z}) + \mathbf{M}_{\sigma_\theta}(\mathbf{z})$, where $\mathbf{M}_{\mu_\theta}(\mathbf{z}) = \mathbf{J}_{\mu_\theta}(\mathbf{z})^\intercal \mathbf{J}_{\mu_\theta}(\mathbf{z})$ and similarly for $\sigma_\theta(\cdot)$. We know from the properties of the determinant that $|\mathbf{M}_{\mu_\theta}(\mathbf{z}) + \mathbf{M}_{\sigma_\theta}(\mathbf{z})| \geq |\mathbf{M}_{\mu_\theta}(\mathbf{z})| + |\mathbf{M}_{\sigma_\theta}(\mathbf{z})|$. This implies that when $|\mathbf{M}_{\sigma_\theta}(\mathbf{z})|$ attains its maximum, the pull-back metric attains a high value as well. Especially, if the $\mu_\theta(\cdot)$ is modeled using bounded activation functions as the $\texttt{tanh}(\cdot)$, which means that potentially $|\mathbf{M}_{\mu_\theta}(\mathbf{z})| \to 0$ if all the first hidden layer becomes close to constant far from the latent codes.

Now, we consider the line $\mathbf{z}(\rho) = \rho \mathbf{v} \in \mathbb{R}^d$ where $\rho \in \mathbb{R}$, which implies that $\mathbf{M}_\theta(\mathbf{z}(\rho))$ is a function $\mathbf{M}_\theta : \mathbb{R} \to \mathbb{R}_{>0}$ with $\mathbf{M}_\sigma(\rho) = \sum_{j=1}^D \sigma_j'(\rho)^2$ where we omit θ to simplify notation. Also, $\sigma_j(\rho) = \gamma_j(\rho)^{-1/2},\ j = 1, \dots, D$ where $\gamma_j(\rho) = \sum_{k=1}^K w_{jk} \kappa_k(\rho) = \mathbf{w}_j^\intercal \boldsymbol{\kappa}(\rho)$ with $\kappa_k(\rho) = \exp(-\lambda_k ||\rho \mathbf{v} - \mu_k||^2)$ and $\mathbf{w}_j \in \mathbb{R}_+,\ k = 1, \dots, K$. Note that $\gamma_j'(\rho) = \sum_{k=1}^K w_{jk} \kappa_k(\rho) 2\lambda_k (\mathbf{v}^\intercal \mu_k - \rho \mathbf{v}^\intercal \mathbf{v})$ and $\gamma_j''(\rho) = \sum_{k=1}^K w_{jk} \kappa_k(\rho) 2\lambda_k [2\lambda_k (\mathbf{v}^\intercal \mu_k - \rho \mathbf{v}^\intercal \mathbf{v})^2 - \mathbf{v}^\intercal \mathbf{v}]$. While we have for the derivative $\sigma_j'(\rho) = -\frac{1}{2} \frac{\gamma_j'(\rho)}{\gamma_j(\rho)^{3/2}}$ that $\lim_{\rho \to \pm\infty} \sigma_j'(\rho)^2 = \lim_{\rho \to \pm\infty} \frac{1}{4} \frac{\gamma_j'(\rho)^2}{\gamma_j(\rho)^3} \to \frac{0}{0}$ due to the RBF. We compute the derivatives of the nominator and denominator and we get that

$$\lim_{\rho \to \pm\infty} \frac{2 \cancel{\gamma_j'(\rho)} \gamma_j''(\rho)}{3 \gamma_j(\rho)^2 \cancel{\gamma_j'(\rho)}} = \lim_{\rho \to \pm\infty} \frac{2 \sum_{k=1}^K w_{jk} \widetilde{\kappa_k}(\rho) 2\lambda_k [2\lambda_k (\mathbf{v}^\intercal \mu_k - \rho \mathbf{v}^\intercal \mathbf{v})^2 - \mathbf{v}^\intercal \mathbf{v}]}{3 \mathbf{w}_j^\intercal \boldsymbol{\kappa}(\rho) \cdot \mathbf{w}_j^\intercal \widetilde{\boldsymbol{\kappa}}(\rho)} \to +\infty, \tag{19}$$

where $\widetilde{\kappa}_k(\rho) = \frac{\kappa_k(\rho)}{\sum_{k=1}^K \kappa_k(\rho)}$. This limit goes to infinity because $0 < \widetilde{\kappa}_k(\rho) < 1$ and $[2\lambda_k (\mathbf{v}^\intercal \mu_k - \rho \mathbf{v}^\intercal \mathbf{v})^2 - \mathbf{v}^\intercal \mathbf{v}] > 0$ when $\rho \to \pm\infty$, while in the denominator $\mathbf{w}_j^\intercal \boldsymbol{\kappa}(\rho) \to 0$. This actually means that the denominator converges to zero faster than then nominator, so the magnitude $\sqrt{|\mathbf{M}_\sigma(\rho)|} \to +\infty$ when the RBF network is close to zero.

Note that the centers and the bandwidths of the kernels are within the support of our prior, as they are trained post-hoc using k-means and by computing the covariance of the corresponding clusters, respectively. So, it holds that $\nu_\psi(\mathbf{z}(\rho)) \to 0 \Rightarrow \gamma_j(\rho) \to 0$, which means that both metrics attain the maximum in the same region of $\mathcal{Z}$. □

Proposition 2. *Let a neighborhood $\mathcal{U} \subset \mathcal{M}$ where $p(\mathbf{x})$ is uniform, a learned $g(\cdot)$ and the proposed prior $\nu_\psi(\mathbf{z})$. If $\sigma_\theta^2(\cdot) \approx \varepsilon$, where $\varepsilon > 0$ a small scalar, then in the corresponding region in $\mathcal{Z}$ the two metrics are related as $\sqrt{|\mathbf{M}_\theta(\cdot)|} = \sqrt{|\mathbf{M}_\psi(\cdot)|}^{-1}$.*

Proof. If $\sigma_\theta^2(\cdot) \approx \varepsilon$ is close to zero within the region $g : \mathcal{Z}_\mathcal{U} \to \mathcal{U} \subset \mathcal{M}$, then we know from Eq. 5 that $\mathbf{M}_\theta(\mathbf{z}) \approx \mathbf{M}_{\mu_\theta}(\cdot)$, since the Jacobian $\mathbf{J}_{\sigma_\theta}(\cdot) \to \mathbf{0}$. This implies that the map $g(\cdot) \approx \mu_\theta(\cdot)$ locally is deterministic, and hence, from the change of variables we know that $p(\mathbf{x}) \sqrt{|\mathbf{M}_\theta(\mathbf{z})|} = \nu_\psi(\mathbf{z})$. Then, if $p(\mathbf{x})$ is uniform and our proposed metric $\mathbf{M}_\psi(\mathbf{z}) = \nu_\psi(\mathbf{z})^{-2/d} \cdot \mathbb{I}_d$ (Eq. 4.1) we get the result $\sqrt{|\mathbf{M}_\theta(\mathbf{z})|} = \sqrt{|\mathbf{M}_\psi(\mathbf{z})|}^{-1},\ \forall \mathbf{z} \in \mathcal{Z}_\mathcal{U}$. □

So, the behavior of the metrics locally is inverse proportional. But if it holds that $m_1 \leq |\mathbf{M}_\psi(\mathbf{z})| \leq m_2$, where $m_1, m_2 > 0$ and $|m_1 - m_2| \to 0$ the metrics do not change significantly within $\mathcal{Z}_\mathcal{U}$. In other words, the curvature of the metrics locally is low, so the behavior of the shortest path is not extremely different. Even in this case though, we cannot guarantee that locally the paths are similar. Since there might exist another neighborhood $\mathcal{U}' \subset \mathcal{M}$ such that $\mathcal{U}' \cap \mathcal{U} \neq \emptyset$, which one of the shortest paths might prefer.

Proposition 3. *Let a neighborhood $\mathcal{U} \subset \mathcal{M}$ where $p(\mathbf{x})$ is uniform. We assume that in the corresponding region in $\mathcal{Z}$ the generator's uncertainty $\sigma_\theta^2(\cdot) \approx \varepsilon$, where $\varepsilon > 0$ a small scalar and that the $\mu_\theta(\cdot)$ has low curvature. Then, for both metrics $\mathbf{M}_\theta(\cdot)$ and $\mathbf{M}_\psi(\cdot)$ the shortest paths within this region are straight lines.*

Proof. From the assumptions we know that $\sigma_\theta(\cdot)$ is approximately constant implying that $\mathbf{J}_{\sigma_\theta}(\cdot)$ goes to $\mathbf{0}_{D \times d}$. The low curvature of $\mu_\theta(\cdot)$ implies that $\mathbf{J}_{\mu_\theta}(\cdot)$ is approximately constant. For example, if the map locally is linear, then it has zero curvature and the $\mathbf{J}_{\mu_\theta}(\cdot)$ is constant. In the general ODE system (see Eq. 9) we use the derivative of the metric. In this case, this quantity will be approximately zero, since the low curvature implies that the metric will not change locally. Therefore, the ODE system becomes $\ddot{c}(t) = 0$, where the solution is the straight line. Note that we solve this ODE as a BVP problem with $c(0)$ and $c(1)$ the given boundary conditions.

Since locally $\sigma_\theta^2(\cdot) \approx \varepsilon$ we know that the mapping is approximately deterministic, and also, $p(\mathbf{x})$ is assumed to be uniform, so $\sqrt{|\mathbf{M}_\theta(\mathbf{z})|} = \sqrt{|\mathbf{M}_\psi(\mathbf{z})|}^{-1}$ (see Prop. 2). However, the curvature of $\mu_\theta(\cdot)$ is by the assumption low,

so $\sqrt{|\mathbf{M}_\theta(\cdot)|}$ is close to constant implying that $\sqrt{|\mathbf{M}_\psi(\mathbf{z})|} = \nu_\psi(\mathbf{z})$ is approximatelly constant as well. Therefore, the ODE system (see Eq. 1) becomes a small perturbation of $\ddot{c}(t) = 0$ and by basic ODE theory the solutions of the two systems are close (e.g. in C^2-norm). Consequently, the solution is again the straight line. □

We are able to provide a more general result that relates any pull-back metric $\mathbf{M}_\theta(\cdot)$ with a conformal metric $\mathbf{M}_\psi(\cdot)$. Briefly, in a local neighborhood let us consider a bounded pull-back metric with small (Riemannian) curvature, i.e. the metric is almost locally isometric to a flat Euclidean space, and the corresponding volume form is tightly controlled. Then, we can reparametrize this neighborhood so that $\mathbf{M}_\theta(\cdot)$ becomes conformally flat, and in addition, we can rescale it such that the magnitude becomes equal to $\sqrt{|\mathbf{M}_\psi(\cdot)|}$.

Proposition D.1. *Let $\mathbf{M}_\theta(\cdot)$ be the Riemannian metric over the latent space $\mathcal{Z}$ as above and suppose that:*

1. *The curvature tensor $R(\cdot)$ associated to $\mathbf{M}_\theta(\cdot)$ satisfies $\|R(\cdot)\|_\infty \leq \kappa$ for a sufficiently small positive κ;*
2. *There exist constants m_1, m_2 with $|m_2 - m_1|$ sufficiently small, so that the volume element $\sqrt{|\mathbf{M}_\theta(\cdot)|}$ satisfies $m_1 \leq \sqrt{|\mathbf{M}_\theta(\mathbf{z})|} \leq m_2$ for each $\mathbf{z}$ in the chart.*

Then there exists a reparametrization $\tilde{\mathbf{z}}$ equipped with the conformal metric $\mathbf{M}_\psi(\tilde{\mathbf{z}})$ (given by a pointwise-diagonal matrix with equal entries whose volume form agrees with $\mathbf{M}_\theta(\mathbf{z})$ at corresponding points as above), so that the geodesics are approximately given by straight lines and locally the volume of the geodesic balls $B_\rho(\mathbf{z})$ and $B_\rho(\tilde{\mathbf{z}})$ are the same.

Proof. Let us suppose w.l.o.g. that the origin $0 \in \mathbb{R}^n$ is contained in the chart and let us consider a normal coordinate neighbourhood $(B_r(0), \overline{\mathbf{z}})$ around 0. It is well-known that in these coordinates the metric tensor $\overline{\mathbf{M}}(\overline{\mathbf{z}})$ at 0 is Euclidean and the following expansion of the metric holds:

$$\overline{M}_{ij}(\overline{\mathbf{z}}) = \delta_{ij} - \frac{1}{3}\overline{R}_{ikjl}(0)\overline{z}_k\overline{z}_l + O(|\overline{\mathbf{z}}|^3), \tag{20}$$

where $\overline{R}$ denotes the respective curvature tensor. In fact, using Jacobi fields one can obtain higher-order expansions whose coefficients are again given by expressions of the curvature tensor. Hence, for every positive number ϵ, if κ is small enough, then $|\overline{M}_{ij}(\overline{\mathbf{z}}) - \delta_{ij}| \leq \epsilon$ for each $\overline{\mathbf{z}} \in B_r(0)$. Moreover, a similar expansion holds for the Christoffel symbols of the Levi-Civita connection induced by $\overline{M}_{ij}$:

$$\Gamma^i_{jk}(\overline{\mathbf{z}}) = -\frac{1}{3}\left(\overline{R}_{ikjl}(0) - \overline{R}_{ijkl}(0)\right)\overline{z}_l + O(|\overline{\mathbf{z}}|^2), \tag{21}$$

hence, for eventually choosing a smaller κ one has $|\Gamma^i_{jk}(\overline{\mathbf{z}})| \leq \epsilon$. Now considering the geodesic equations for $\overline{\mathbf{M}}(\overline{\mathbf{z}})$ and the flat Euclidean metric at $\mathbf{M}_e(\overline{\mathbf{z}})$, by basic perturbation theory for ODEs, one sees that the solutions (i.e. the geodesic curves) within $B_r(0)$ satisfy:

$$\|\gamma_{\mathbf{M}_e} - \gamma_{\overline{\mathbf{M}}}\|_{C^\infty} \leq \epsilon, \tag{22}$$

if κ is chosen small enough (shrunk further). This implies that the geodesic distances induced by the Euclidean metric $\mathbf{M}_e(\overline{\mathbf{z}})$ and $\overline{\mathbf{M}}(\mathbf{z})$ are similar i.e., straight lines.

Finally, let us pointwise rescale the coordinates $\overline{\mathbf{z}}$ to $\tilde{\mathbf{z}}$ so that the metric $\mathbf{M}_e(\overline{\mathbf{z}})$ agrees with the conformal metric $\mathbf{M}_\psi(\tilde{\mathbf{z}})$ - in particular, the volume forms at $\tilde{\mathbf{z}}$ and $\mathbf{z}$ agree (since we assumed that $|m_2 - m_1|$ is sufficiently small, the rescale is essentially given by constant multiplication). Here $(\tilde{\mathbf{z}}, \mathbf{M}_\psi(\tilde{\mathbf{z}}))$ is the conformal metric as defined above. Moreover, by construction the volumes of the geodesic balls $B_\rho(\tilde{\mathbf{z}})$ and $B_\rho(\mathbf{z})$ agree. □

Obviously, the Prop. D.1 implies that if $\mathcal{Z}$ is the latent space where the pull-back metric is defined, if we reparametrize it to get an equivalent conformally flat metric we work over a new space $\mathcal{Z}'$. This is not easily applicable and useful in our setting, since we are interested to compute shortest paths directly in the latent space $\mathcal{Z}$ of the generative model. However, when Prop. 3 applies, then the reparametrization from $\mathcal{Z}$ to $\mathcal{Z}'$ is actually the indentity map. Also, we are able to rescale the magnitude such that the maximum value on the training latent codes is the same (see App. B). This result implies that shortest paths are approximately straight lines and with equal curve length under each corresponding metric. For a demonstration see App. E.3.

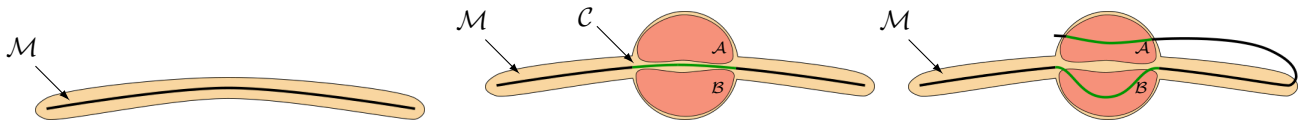


Figure 7: *Left*: A data manifold where the data are uniformly distributed $p(\mathbf{x})$ near $\mathcal{M}$. *Middle*: A ball of uniform samples is located in the center of $\mathcal{M}$. The mean function $\mu_\theta(\cdot)$ of the generator $g(\cdot)$ approximates the regions $\mathcal{A}$ and $\mathcal{B}$ with the region $\mathcal{C}$, while the corresponding uncertainty $\sigma_\theta(\cdot)$ is high. *Right*: A rather unrealistic scenario of approximating the data manifold with a VAE.

Proposition 4. *Let a data manifold $\mathcal{M}$ where $p(\mathbf{x})$ is uniform. We assume that there is a neighborhood $\mathcal{U} \subset \mathcal{M}$ where the intrinsic dimensionality increases. Then, in the corresponding region of $\mathcal{Z}$ the two metrics $\mathbf{M}_\theta(\cdot)$, $\mathbf{M}_\psi(\cdot)$ have exactly the opposite behavior.*

Proof. We need to show that: 1) the prior in the latent space increases in the region which corresponds to the reconstruction of the neighborhood $\mathcal{U}$ where the dimensionality of the data manifold increases, and 2) the uncertainty of the generator $\sigma_\theta(\cdot)$ in the same region increases as well. We provide an illustration in Fig. 7, where the dimensionality of the data manifold locally increases from 1 to 2.

The ELBO is maximized with respect to the parameters of the functions and we consider each term individually:

1. $\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] = \mathbb{E}_{p(\varepsilon)}\left[\sum_{j=1}^{D} \log \sigma^2_{\theta,j}(\mu_\phi(\mathbf{x}) + \sigma_\phi(\mathbf{x}) \odot \varepsilon) + \frac{||\mathbf{x} - \mu_\theta(\mu_\phi(\mathbf{x}) + \sigma_\phi(\mathbf{x}) \odot \varepsilon)||^2}{\sigma^2_\theta(\mu_\phi(\mathbf{x}) + \sigma_\phi(\mathbf{x}) \odot \varepsilon)}\right]$. Due to the Gaussian likelihood and the L2-norm, the mean function of the generator $\mu_\theta(\cdot)$ approximates the points locally as a "mean-value". Also, this term pushes implicitly $\sigma_\phi(\cdot)$ to low values to be able to reconstruct the point $\mathbf{x}$ as good as possible. Since the overlapping region between multiple distributions $q_\phi(\mathbf{z}|\mathbf{x}_j)$, $j = 1, \ldots, J$ results to a $p(\mathbf{x}|\mathbf{z})$ with low accuracy such that to approximate well all the points $\mathbf{x}_j$ simultaneously.
2. $\mathrm{KL}[q_\phi(\mathbf{z}|\mathbf{x})||\mathcal{N}(0, \mathbb{I}_d)]$. This term acts as a mild regularizer that pushes $\sigma_\phi(\cdot) \to 1$ and $\mu_\phi(\cdot) \to 0$.
3. $\mathbb{E}_{p(\varepsilon)}[f_\psi(\mu_\phi(\mathbf{x}) + \sigma_\phi(\mathbf{x}) \odot \varepsilon)]$, tries to maximize the function $f_\psi(\cdot)$ on the latent codes. So implicitly the $\sigma_\phi(\cdot)$ needs to be small, such that the encoded samples to result in high $f_\psi(\cdot)$ values.
4. $-\log(\mathcal{C})$ regularizes the function $f_\psi(\cdot)$ to be small far from the training latent codes, which implicitly forces $\sigma_\phi(\cdot)$ to be small, such that to keep the support of the encoder within the high values of the prior.

From the analysis above we see that the encoder tends to have negligible variance, which is a known result [Dai and Wipf, 2019]. In addition, we know that under the Gaussian likelihood the generator $g(\cdot)$ favours approximations of the data manifold in an L2 sense as illustrated in Fig. 7 (left, middle). Let $\mathcal{Z}_\mathcal{A}, \mathcal{Z}_\mathcal{B}$ be the regions in $\mathcal{Z}$ where the encoder maps the regions $\mathcal{A}$ and $\mathcal{B}$. Then, we have that $\mathcal{Z}_\mathcal{A} \approx \mathcal{Z}_\mathcal{B}$ since $\mathcal{C} \approx \mu_\theta(\mathcal{Z}_\mathcal{A}) \approx \mu_\theta(\mathcal{Z}_\mathcal{B})$. Note that $\sigma_\theta(\cdot)$ is high in the same region of the latent space. In a different scenario where $\mathcal{Z}_\mathcal{A} \cap \mathcal{Z}_\mathcal{B} = \emptyset$ the $\mathcal{A}, \mathcal{B}$ may be approximated as depicted in Fig. 7 (right). However, we argue that this type of complex VAE solutions are not achievable in practice, because d is typically much smaller than D.

The aggregated posterior is written as $q(\mathbf{z}) = \int_\mathcal{X} q_\phi(\mathbf{z}|\mathbf{x})p(\mathbf{x})d\mathbf{x} \approx \frac{1}{N}\sum_{n=1}^{N} q_\phi(\mathbf{z}|\mathbf{x}_n)$, where $\mathbf{x}_n$ the training data, and it has been shown in Tomczak and Welling [2018] that this is the optimal prior. We also know that the variance of the encoder is in general small and that $\mathcal{Z}_\mathcal{A} \approx \mathcal{Z}_\mathcal{B}$. This implies that $q(\mathbf{z})$ is high in the corresponding region due to the high number of encoded points. Also, note that our learnable prior approximates the $q(\mathbf{z})$. Consequently, the proposed metric $\mathbf{M}_\psi(\cdot)$ is small in the latent region $\mathcal{Z}_\mathcal{A} \approx \mathcal{Z}_\mathcal{B}$.

In contrast, the $\sigma_\theta(\cdot)$ is high in this region. Let us distinguish two sets of kernels for the RBF: The set S_1 for which the kernel weights w_k are big (low $\sigma_\theta(\cdot)$) and the set S_2 where the weights w_k are small (high $\sigma_\theta(\cdot)$). We then consider two functions $f_1(\mathbf{z}) = \sum_{k \in S_1} w_k \kappa_k(\mathbf{z})$ and $f_2(\mathbf{z}) = \sum_{k \in S_2} w_k \kappa_k(\mathbf{z})$. The $f_1(\cdot) > \varepsilon$ in regions with low variance and $f_2(\cdot) < \varepsilon$ in regions with high variance, where $\varepsilon > 0$. So the Jacobian of $\sigma_\theta(\cdot)$ is non-zero in the set $S = \{\mathbf{z} \mid f_1(\mathbf{z}) = f_2(\mathbf{z}) \text{ and } \nu_\psi(\mathbf{z}) > 0\}$ as we go from kernels that correspond to low variance $\sigma^2_\theta(\cdot)$ to kernels with high $\sigma^2_\theta(\cdot)$. Therefore, the pull-back metric $\mathbf{M}_\theta(\cdot)$ is high in this regions. □

We provided a theoretical analysis of our metric compared to the pull-back. Also, we discussed some problematic cases that might occur in practice, while additional problematic cases may exist. However, Prop. 1 shows that "globally" the behavior of the shortest paths is similar, while it depends only on mild assumptions.

E Experimental details

In this section, we give the details about our experimental setting and implementation, as well as additional demonstrations. The source code can be found https://github.com/georgiosarvanitidis/geometric_ml.

E.1 Shortest path solver

One of the main tools that we use in our experiments is the solver for ODE system. Specialized approximate numerical solvers have been proposed [Hennig and Hauberg, 2014; Arvanitidis et al., 2019; Yang et al., 2018] mainly aiming for efficiency. However, usually the off-the-self numerical solvers provide more accurate solutions, especially as regards the logarithmic maps, and for this reason our approach is based on the SciPy's BVP solver. Commonly, solvers of this type implement a version of Newton's method, and thus, convergence heavily relies on the initial solution. For this reason, we use a heuristic graph based solver, in order to provide a curve to the BVP solver as a good initial solution.

For the heuristic solution, we construct a k-nearest neighbor graph in the latent space $\mathcal{Z}$ by using the Euclidean distances to find the neighbors. Once we construct the graph, we assign as edge weights the length of the straight line computed under the Riemannian metric. Essentially, a large edge weight informs us that this is not a good connection. For two test points, we find their k-NN neighbors on the graph again using the Euclidean metric first, and then, we update the weights of the edges using the Riemannian metric. This step enables us to use as the starting and ending point, the nodes of the graph that are closer to the test points. Then, we can find the discrete shortest path on the weighted graph using Dijkstra's algorithm. Note that the algorithm runs only once and the pairwise paths are stored.

This returns a sequence of points, but for which the starting and ending points are not the actual test points. So once we have this sequence, we replace the two points on the boundaries with the test points. In order to smooth the final path we apply a simple filter. We update each point except the ones at the boundaries as $\mathbf{p}_i = (\mathbf{p}_{i-1} + \mathbf{p}_i + \mathbf{p}_{i+1})/3$. Of course, we can apply more sophisticated filtering techniques. Finally, we use a *cubic spline* to interpolate the filtered points, including the boundary test points.

Clearly, this is a heuristic solution and does not satisfy the corresponding ODE system. However, in many cases it constitutes a sufficiently good initial solution, which helps the BVP solver to converge. So for the solver that we use, if we do not have a previous solution computed, we initialize the solver with the heuristic graph based curve. If the BVP solver fails, then we return the graph based curve as our solution. However, in cases where the logarithmic map is necessary (as the LAND fitting) and the BVP solver fails, we exclude the point from the current step of the algorithm. The reason is that the logarithmic map of the graph based curve is totally arbitrary, and thus, we do not use it.

E.2 Details for the comparison of the priors

Here we present the details for our generative modeling experiment where we compare three priors in a VAE setting. In particular, we used two settings, a Convolutional VAE and a standard VAE. In Table 2 and Table 3 we present the details of each setting. Note that for the standard VAE we projected the data using PCA in 100 dimensions with whittening, so the given data are in $\mathcal{X} = \mathbb{R}^{100}$. This step keeps $> 90\%$ of the data variance and on the same time allows to train an RBF network post-hoc in order to induce the pull-back metric in $\mathcal{Z}$. We used the standard data splitting train/test both for MNIST and FashionMNIST[1].

For the RBF network that models $\sigma_\theta(\cdot)$ we use $K = 100$ components. The RBF network is fitted post-hoc. Specifically, in the first phase of the VAE training we use a deep neural network to model $\overline{\sigma}^2_\theta(\cdot)$ and in the second phase we train individually the RBF's weights i.e. as a regression problem or using the ELBO keeping the rest of the functions fixed. The RBF's centers are trained with k-means using the training latent codes. Then, using the points in each cluster we compute the corresponding covariance matrices, and then, for the bandwidth of each kernel we use the minimum variance on the diagonal of each covariance. This approach guarantees that the centers are near the latent codes, while the bandwidths are small enough such that to capture precisely the structure of the latent codes and be in the support of $\nu_\psi(\cdot)$. Note that training the RBF together with the VAE in one phase might move centers far from the latent codes. Also, it is hard to pre-specify the bandwidth of the

[1] https://pytorch.org/docs/stable/torchvision/datasets.html

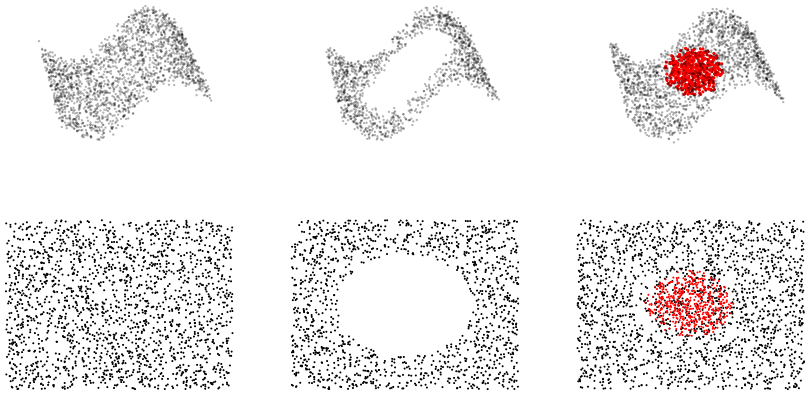


Figure 8: The synthetic data for the constructive examples (normal, hole, ball).

$\text{encoder}_\phi(\cdot)$	$3 \times 3 \times 32$	$3 \times 3 \times 32$	$3 \times 3 \times 32$
$\mu_\phi(\cdot)$	$\text{encoder}_\phi(\cdot)$	flatten	MLP(512, d)
$\log(\sigma^2_\phi(\cdot))$	$\text{encoder}_\phi(\cdot)$	flatten	MLP(512, d)
$\mu_\theta(\cdot)$	MLP(d, 512)	unflatten	$\text{decoder}_\theta(\cdot)$
$\text{decoder}_\theta(\cdot)$	$4 \times 4 \times 32$	$4 \times 4 \times 32$	$4 \times 4 \times 32$

Table 2: The convolutional VAE details. Encoder: We used convolutional filters with padding 1 only in the first two filters, stride 2 only in the first 2 filters and 1 in the final filter. Decoder: We used transposed convolutions with padding 1 only in the last two filters, stride 2 only in the last 2 filters and 1 in the fist filter. Also, we applied a final convolution filter $3 \times 3 \times 1$ with stride 1 and padding 1 to provide a smooth output. We used `tanh` activations both layers, convolutional and MLP.

kernels, and if this parameter is trainable usually overestimated bandwidths occur. For the VampPrior we used $K = 500$ trainable inducing points.

We trained all the parameters using the Adam optimizer with learning rate $1e^{-3}$. The batch size is 128 and the number of epochs is 500. For our prior we estimate the normalization constant using 10·batch_size samples from $p(\mathbf{z})$, and in addition we included the latent codes of the training data in the batch. This helps to regularize the behavior of $f_\psi(\cdot)$, such that to prevent extreme values on the latent codes.

E.3 Details for the constructive examples

We construct a surface in $\mathcal{X} = \mathbb{R}^3$ with $\mathbf{x} = [\mathbf{z}, 0.25 \cdot \sin(z_1)] + \varepsilon$ where $z_j \sim \mathcal{U}(0, 2\pi),\ j = 1, 2$ and $\varepsilon \sim \mathcal{N}(0, 0.1^2 \cdot \mathbb{I}_3)$. We call this as the normal dataset. We also construct a surface with a hole in the middle by removing the points in the center with radius $||\mathbf{z}||_2 < 0.3$, before the mapping in $\mathbb{R}^3$. Finally, we construct a uniform ball directly in $\mathbb{R}^3$ with radius $||\mathbf{x}||_2 < 0.2$ that we place in the center of the normal surface. We present the datasets in $\mathcal{X}$ and the corresponding true latent codes in Fig. 8. These are the three datasets that correspond to the analysis we did in Sec. 4. In fact, the normal surface is the closest one to Prop. 3, as the manifold has low curvature locally.

For the deep neural networks and the RBF we use the same setting as in App. E.2. Also, we use for the solution of the ODE system the strategy presented in App. E.1. As we note in the main paper (see Sec. 5) we reparametrize the curves with respect to the Euclidean metric. This allows to compare as good as possible the actual curves in $\mathcal{Z}$. In this way, we can compare how "close" are the two curves in the space.

Additionally, we show in Fig. 9 and Fig. 10 a second comparison to demonstrate Prop. D.1. Basically, we compare the actual lengths of the shortest paths computed under each Riemannian metric. We train a VAE and an RBF using the normal surface data. In the first Fig. 9 the bandwidth of the RBF kernels is scaled by 1.5, which makes the uncertainty term of the pull-back metric (second term in Eq. 5) close to zero. In other words, this

$\text{encoder}_\phi(\cdot)$	MLP(D, H)	MLP(H, H)
$\mu_\phi(\cdot)$	$\text{encoder}_\phi(\cdot)$	MLP(H, d)
$\log(\sigma^2_\phi(\cdot))$	$\text{encoder}_\phi(\cdot)$	MLP(H, d)
$\text{decoder}_\theta(\cdot)$	MLP(d, H)	MLP(H, H)
$\mu_\theta(\cdot)$	$\text{decoder}_\theta(\cdot)$	MLP(H, D)
$\log(\sigma^2_\theta(\cdot))$	$\text{decoder}_\theta(\cdot)$	MLP(H, D)

Table 3: The VAE encoder details. We used `tanh` activations for the MLPs.

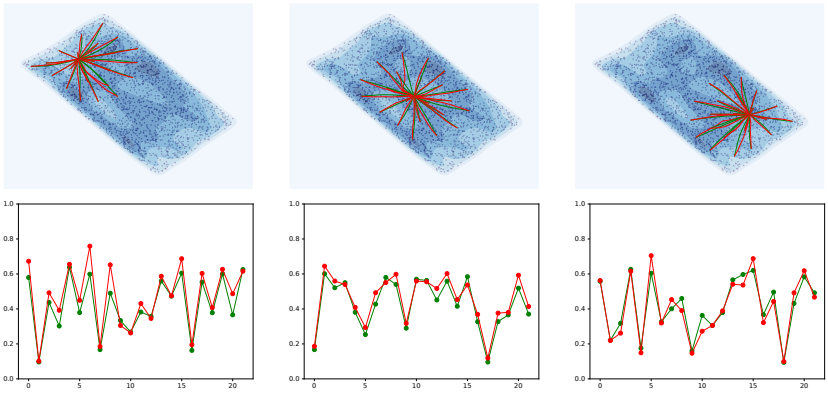


Figure 9: *Top row*: Comparing the shortest path distance computed under each metric $\mathbf{M}_\psi(\cdot)$ with green and $\mathbf{M}_\theta(\cdot)$ with red. When conditions in Prop. 3 hold, then shortest paths are approximately straight lines. The scaled metrics have maximum magnitude 1 on the training latent codes, so the curve lengths are approximately equal. *Bottom row*: Each dot in the graphs corresponds to a curve length, while the connecting lines are shown for better illustration.

implies that the $\sigma_\theta(\cdot)$ is nearly constant. Since this is a simple surface the curvature of $\mu_\theta(\cdot)$ is expected to be low. Additionally, we expect the encoder to provide a nearly uniform distribution in $\mathcal{Z}$ since the data are almost uniformly distributed in $\mathcal{X}$. Note that we rescale the metrics (see App. B) such that the highest mangification factor on the training latent codes to be 1 in the neighborhood of $\mathcal{Z}$ that we consider. Hence, as expected by Prop. D.1 both metrics result to shortest paths that have approximately equal lengths (Fig. 9). However, when the bandwidth of the RBF is not scaled by 1.5, it is very small, so the second term of the pull-back changes fast. Hence, even if $\mu_\theta(\cdot)$ remains the same, the uncertainty term increases the curvature. Consequently, the lengths are not similar anymore (Fig. 10). We could potentially use always larger bandwidths for the kernels to alleviate this issue. The problem then is that we lose the locality of the RBF, which implies that we do not capture precisely the structure of the data manifold. In other words, we will allow the shortest paths to move in regions of $\mathcal{Z}$ with no latent codes, which does not necessarily correspond to the data manifold in $\mathcal{X}$.

E.4 Details for efficiency and robustness

For the experiments that we conducted in Sec. 5 we used the same VAE setting as in App. E.2 and we projected using PCA in 100 dimensions the MNIST digits 0,1,2.

For having comparable magnitudes we scale each metric so that the highest magnitude on the training latent codes is 1 (see App. B). For our proposed metric we also set the upper bound to be 100. As regards the pull-back metric we cannot explicitly control the upper bound of the magnitude. So we set ζ in the RBF such that the maximum $\sigma^2_\theta(\cdot)$ to be 1000 times the mean variance of the training latent codes (see App. B).

Here we explain in more details the result in Fig. 6. In higher dimensions the RBF network for some points can easily give a very high $\mathbf{J}_{\sigma_\theta}(\cdot)$. These points lie on the boundary of the kernels where the $\sigma^2_\theta(\cdot)$ changes extremely fast. More specifically, this occurs in the "tails" of the RBF kernel. As we know due to the curse of dimensionality this phenomenon is common for the kernels in high dimensions. Therefore, some of the latent codes may obtain a very high magnitude, which means that this will scale down significantly the metric (see

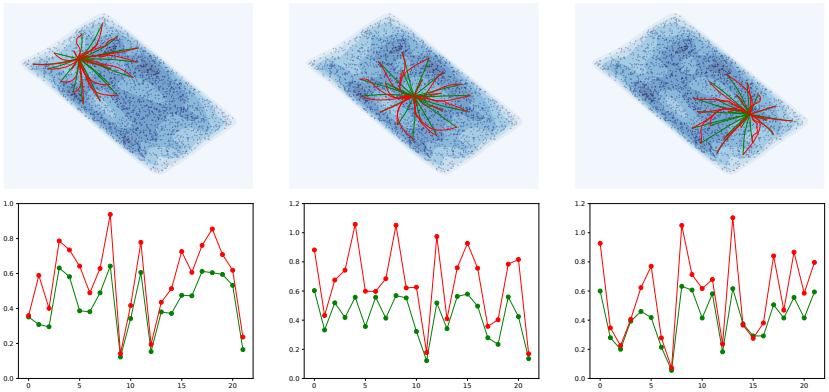


Figure 10: *Top row*: Comparing the shortest path distance computed under each metric $\mathbf{M}_\psi(\cdot)$ with green and $\mathbf{M}_\theta(\cdot)$ with red. When conditions in Prop. 3 does not hold, then shortest paths are not straight lines. However, when the points are very close then the paths are similar. Note that the metrics are scaled to be maximum 1 on the training latent codes. *Bottom row*: Each dot in the graphs correspond to the length of a curve, while the connecting lines only for better illustration.

Eq. 15). This is precisely what we observe in Fig. 6 (*right*) for $d = 10$. In this case the distribution of the magnitudes has two modes. The interpretation is that there are some latent codes near the centers and within the support of the RBFs where the magnitude is normal, while there are few points closer to the tails causing a huge downscaling to the normal magnitudes.

As regards our metric, its behavior is robust, which means that the prior near the latent codes is non-zero and the actual density values across them are comparable i.e. $\nu_\psi(\cdot)$ is close to uniform. Also, there are no training latent codes that get extreme prior values e.g. close to zero. In order to show that only near the latent codes the magnitude is small, we sampled uniformly from the hypercube that surrounds the training codes and evaluated the metric. The result in Fig. 11 shows that, indeed, only near the latent codes the metric is small. Especially, as the dimension increases, there is more empty space in the hypercube with no latent codes so the magnitude is large. The interpretation is that the proposed prior adapts well on the training latent codes and does not assign density in regions of $\mathcal{Z}$ with no codes. Of course, an extremely flexible prior should not be used, as this could easily overfit the latent codes. This implies a highly curved metric i.e., the metric changes extremely fast.

Finally, due to the huge curvature of the pull-back metric, mainly due to the uncertainty term, the shortest path solver has to run for longer, while it fails many times. Note that even the evaluation of the metric and its derivative that we need for the ODE system (see Eq. 9) is significantly more computationally expensive compared to our proposed metric. Here we used a different strategy to compute shortest paths. First we run the BVP solver with the straight line as the initial solution, and if this fails, we re-run the solver initialized by the graph based solution. The reason for doing that is to show that the ODE system under the proposed metric is easier and it can be solved directly without the graph initialization.

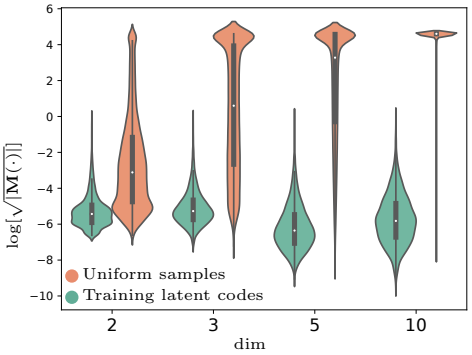


Figure 11: Comparing the magnitude between the training latent codes and uniform samples in the bounding box.

E.5 Details for the LANDs experiment

For the LAND experiment we used the same VAE, RBF and data as in App. E.4. Note that we did not use all the latent codes for training the LANDs, but instead, we quantized them using k-means with 120 centers. Even if the proposed metric is much more efficient than the pull-back, still computing one shortest path relies on the solution of an ODE system. This makes the use of all the latent codes rather prohibited. As reported in the main paper (see Sec. 5) the running times are significantly different, as it is much more efficient to compute shortest paths under our proposed metric.

Moreover, we observe that the pull-back metric underestimates the precision matrices (or overestimates the covariance matrices). The reason is that for some points the shortest path length is large, because the pull-back metric gets large due to the non-robustness of the RBF term. So the corresponding logarithmic map is large as well. This causes the precision to become smaller such that to capture these points that lie "far" from the component's center. Obviously, this is not a desirable behavior, since it only occurs due to the poor behavior and non-robustness of the RBF.

E.6 Details for for life science experiments

Here we explain the details for the experiments with the real-world datasets (see Sec. 5). As we mention in the main paper these experiments should be considered as a proof-of-concept, since specialized generative models have been proposed in the literature for this type of data. With our experiment we want to show that geometry might be a suitable theory to utilize for exploratory data analysis in life sciences.

Mouse cortex cell data. For the mouse cortex cell data [Zeisel et al., 2015] we used the scvi-tools (https://www.scvi-tools.org). As a reprocessing step we kept the 558 genes with the highest variability, and we projected the data into 100 dimensions using PCA for simplicity. For the VAE and the RBF we used the setting as App. E.2. For the shortest path solver we used the setting as in App. E.1. Since for the LAND fit we need the logarithmic maps to be as precise as possible, if the BVP solver fails, then we do not consider this point for the corresponding mixture component. Also, we quantized the latent codes using k-means with $k = 200$.

The resulting LAND adapts better to the latent codes, especially when we observe the individual components (Fig. 14). Interestingly, even the centers between the GMM and the mixture of LANDs differ. One reason is that the Euclidean distance of the latent codes does not correspond to the actual distance on the data manifold. For example, if some points are very sparse on the data manifold in $\mathcal{X}$, the encoder will push everything towards the support of the base distribution $p(\mathbf{z})$ of our proposed prior. However, the Euclidean mean in $\mathcal{Z}$ is not aware of the data manifold's geometry in $\mathcal{X}$, while the LAND mean potentially corresponds to a better estimate on the actual data manifold. In addition, the geometry aware mean under our proposed metric will be closer to the high density region in $\mathcal{Z}$. For instance, assume that the latent codes exhibit a non-convex distribution as a semi-circle. In this case, the Euclidean mean will be outside of the latent codes support, while our mean will be in the support and in particular in the center of the semi-circle.

In addition, we show that we can use the principal geodesics for each component (see Fig. 12), which can be seen as a form of local *disentaglement*. Specifically, we eigen-decompose the covariance matrices of the LANDs mixture and we solve the exponential map with initial velocities the eigen-vectors. Clearly, the resulting paths correspond to the directions with the highest variance on the data manifold in $\mathcal{X}$. In this way we can recover locally the intrinsic degrees of freedom of the data. This can be seen as a form of non-linear PCA. Geometry aware disentanglement seems as a promising direction for future research [Pfau et al., 2020].

Chemical compounds. We used the ZINC database (https://zinc.docking.org/) [Sterling and Irwin, 2015] with SMILES representation, we sampled randomly 6400 points and for simplicity we kept only the first 30 characters of each sequence. Clearly, this is a rather simplified setting, however, patterns of chemical structures are still present. Each batch has dimension $128 \times 30 \times 1$, using one-hot encoding. For the VAE we used an encoder based on 1-dimensional Convolutions and a recurrent decoder (see Table 4).

$\text{encoder}_\phi(\cdot)$	5×32	5×32	5×32
$\mu_\phi(\cdot)$	$\text{encoder}_\phi(\cdot)$	flatten	MLP(128, d)
$\log(\sigma^2_\phi(\cdot))$	$\text{encoder}_\phi(\cdot)$	flatten	MLP(128, d)
$\text{c}_\theta(\cdot)$	MLP(d, H)		
$\text{hs}_\theta(\cdot)$	MLP(d, H)		
$\text{decoder}_\theta(\cdot)$	LSTM(H, $\text{c}_\theta(\cdot)$, $\text{hs}_\theta(\cdot)$)		MLP(H, 1)

Table 4: Recurrent VAE. Enc: Conv1d with kernel size 5 and 32 filters, stride 1 in the first conv and stride 2 for the other two. Dec: For the output sequence we input 1 at each LSTM step and get the output value pushing the hidden state through an MLP. We used `tanh` activations and $H = 128$.

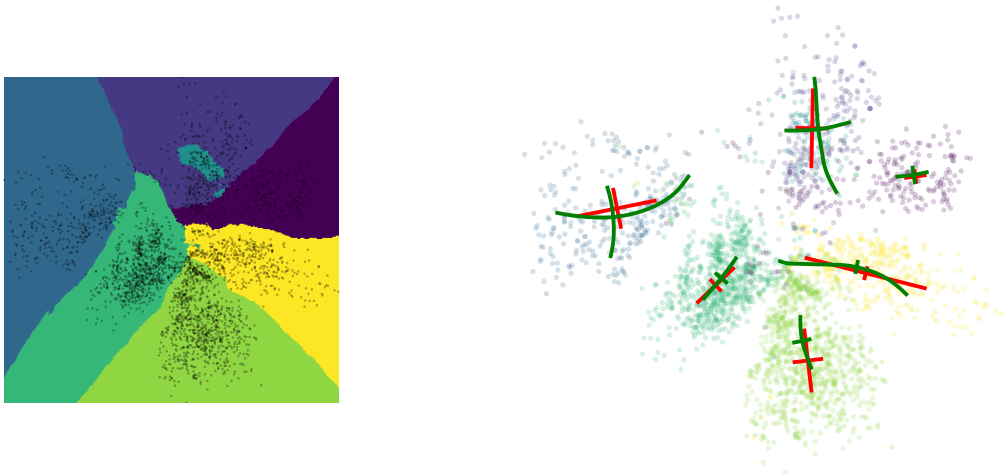
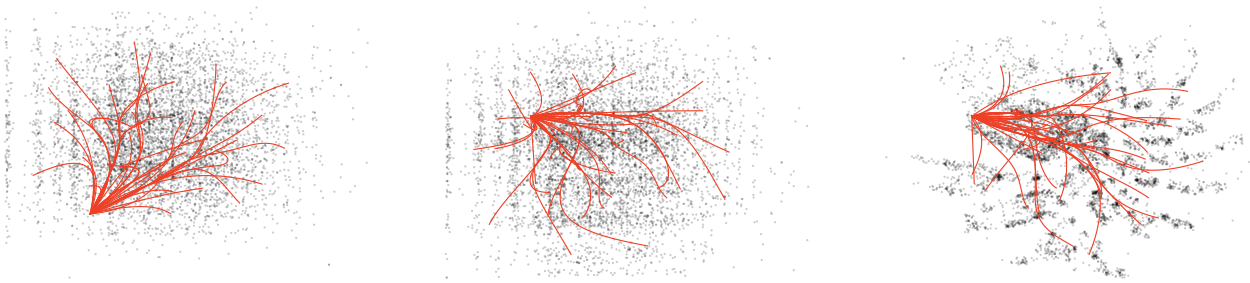


Figure 12: Cortex dataset. *Left*: The true clusters of the latent codes in $\mathcal{Z}$. We used a k-NN classifier to approximate the true clusters with $k = 21$. *Right*: Comparing the principal geodesics computed using the mixture of LANDs (*green*) with the linear eigenvectors computed using the GMM (*red*) for each component. The principal geodesic can be seen as a form of *local geometric disentaglement*, since these paths correspond to the highest variance on the data manifold.

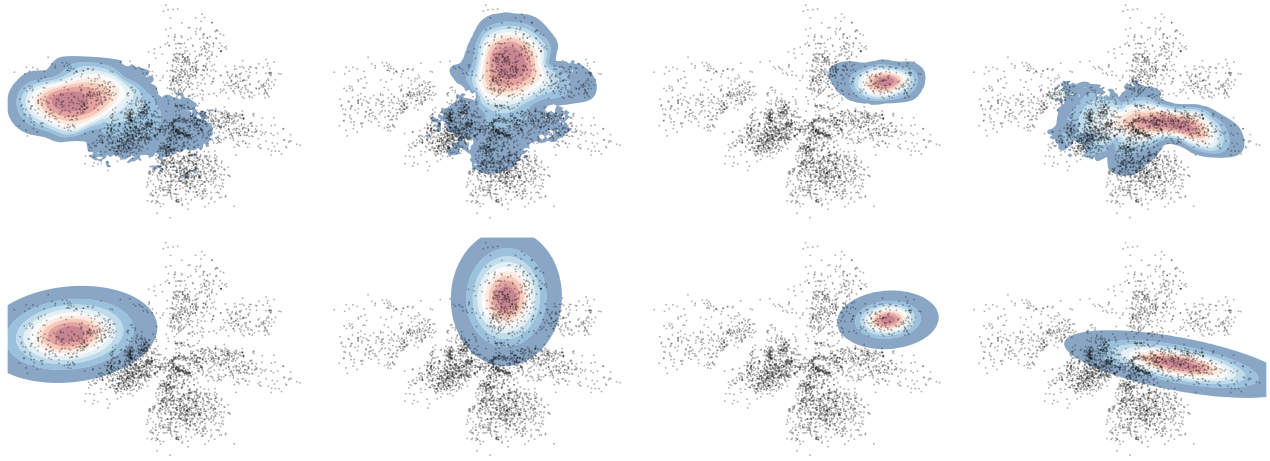
Chemical compounds have by definition an inherent natural structure. As we observe in Fig. 13 even if our setting is rather simplified the resulting representations are indeed non-linearly structured, which our shortest paths respect. In this way, we are able to find interpretable and more meaningful interpolants between points, we can compute mean values and barycenters accordingly, etc. Hence, geometry could potentially uncover some useful properties in the latent space.

To quantify the performance, we measure the cosine similarity in the one-hot encoding space of the generated points along two interpolants (straight line vs our shortest path). What we observe is that our shortest paths generate diverse samples as they follow the nonlinear structure of the representations. In contrast, the straight line does not take into account this structure and crosses regions of $\mathcal{Z}$ with no latent codes, where the behavior of the generator is either constant or arbitrary. We see this in Fig. 5, where for our paths the similarity is high only within small time intervals. Instead, the line crosses regions in $\mathcal{Z}$ with near zero density and generates rather fuzzy samples. We argue that this might not be a useful and sensible behavior, since this kind of data has some biological properties that we need to respect. For instance, the generated samples from the interpolants should be more explicit in order to determine clearly the local properties encoded in the latent space i.e. we want to avoid zero density areas where the generator's behavior is arbitrary. Also, we may want with the interpolation to explore efficiently the latent space and the sequence of the samples should be biologically meaningful, which we cannot guarantee with the straight line. In this spirit, Detlefsen et al. [2020] studied the behavior of latent representations for protein sequences and showed that structure aware paths reveal biological information that is otherwise obscured. Note that our metric allows to capture the structure of the data manifold in the latent space, even if the generator $g(\cdot)$ is not suitable such that to induce a pull-back metric.

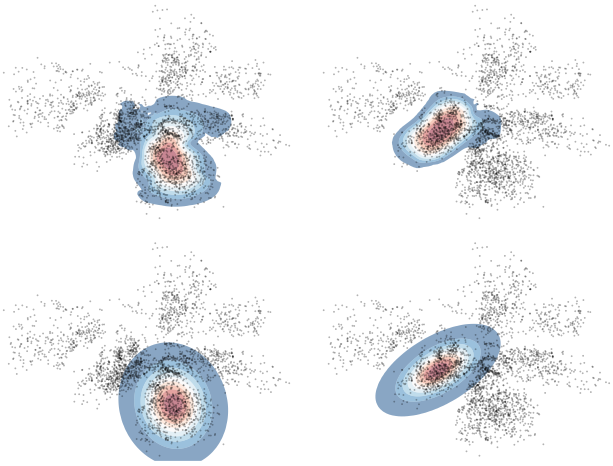


(a) View xy-axis (b) View xz-axis (c) View yz-axis

Figure 13: The latent space of the molecule experiment from different views.



(a) Cluster 1 (b) Cluster 2 (c) Cluster 3 (d) Cluster 4



(e) Cluster 5 (f) Cluster 6

Figure 14: Cortex dataset. The individual components for the mixture of LANDs and the corresponding Gaussian mixture model. *Top row*: The LAND component. *Bottom row*: The corresponding GMM component. We see that the LAND components adapt to the training latent codes, uncovering the structure for each cluster.