Can VAEs capture topological properties?

Luis A. Pérez Rey l.a.perez.rey@tue.nl
Vlado Menkovski v.menkovski@tue.nl
Jacobus W. Portegies j.w.portegies@tue.nl

Abstract

To what extent can Variational Autoencoders (VAEs) identify semantically meaningful latent variables? Can they at least capture the correct topology if ground-truth latent variables are known? To investigate these questions, we introduce the Diffusion VAE, which allows for arbitrary (closed) manifolds in latent space. A Diffusion VAE uses transition kernels of Brownian motion on the manifold. In particular, it uses properties of the Brownian motion to implement the reparametrization trick and fast approximations to the KL divergence. We show that the Diffusion Variational Autoencoder is indeed capable of capturing topological properties.

1 Motivation

A large part of unsupervised learning is devoted to the extraction of meaningful latent factors that explain a certain data set. The terminology around Variational Autoencoders (VAEs) [1, 2] suggests that they are a good tool for this task: they encode datapoints in a space that is called *latent space*. Purely based on this terminology, one could be tempted to think of elements in this space as latent variables, but **is this interpretation warranted?**

As an example, one could think of pictures of different objects, rotated over various angles, translated over various positions, taken under different lighting conditions. Desired algorithms should discover angles, positions and lighting conditions as latent variables.

One aspect of such algorithms that currently receives a lot of attention is the *disentanglement of latent factors*. It means that a desired algorithm should cleanly separate (disentangle) the various latent factors: angles should be separated from the type of the object and from lighting. A variety of algorithms applied to this end are based on VAEs and have a linear latent space [3, 4, 5, 6, 7]; separation should correspond to latent factors being represented in independent subspaces.

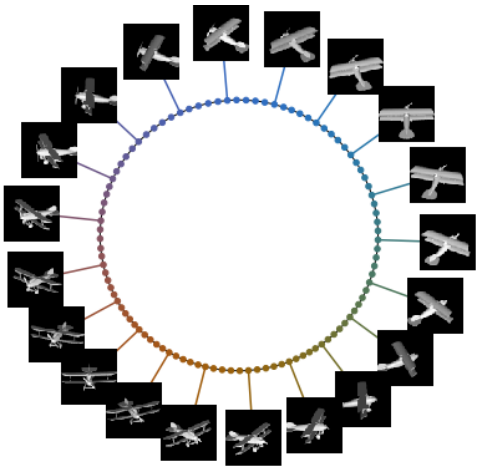


Figure 1: Latent space and reconstruction images for a ΔVAEs with S^1 as latent space trained on the rendered images of a 3D model of a rotating airplane. The ΔVAE manages to capture the underlying geometrical structure.

But there is a very important, different aspect to such algorithms as well, that is still relevant even when a factor such as rotation is disentangled from the rest. The rotation of an object is, due to its periodicity not a linear latent factor, and cannot be disentangled into linear subspaces further. Still, we want our algorithm to somehow capture it, but what do we even mean by that?

4th workshop on Bayesian Deep Learning (NeurIPS 2019), Vancouver, Canada.

To be precise, a rotation *could*, in principle, satisfy the definition of linear disentanglement by Higgins et al. [8]. However, simple experiments with rotations of objects show that in general, it does not in case of a simple VAE [9]. This illustrates the need for either a different definition of "capturing a rotation", or a different VAE. We follow both directions.

We can sharpen the discussion by assuming that we know a ground-truth generative process for the pictures. For instance, pictures are generated by picking at random an angle α in the circle S^1 which we call Z_{true}, rotating an airplane by that angle and taking a picture $\mathsf{Gen}(\alpha)$.

Suppose for the moment that we are working with a VAE, we give the latent space also the structure of S^1 and call it Z_{VAE}. The encoder Enc of the VAE would take a picture as input and produce an element in Z_{VAE} as output.

Naively, from the element in latent space we would like to be able to read off directly what was the original angle. However, since rotations of latent space do not change the loss of the autoencoder, this would be too much to ask for (the choice of origin is somehow arbitrary and can never be found by the autoencoder). The best we can hope for is that the composition

$$Z_{\mathsf{true}} \xrightarrow{\mathsf{Gen}} X \xrightarrow{\mathsf{Enc}} Z_{\mathsf{VAE}}$$

is an isometry, i.e. it is a bijection that preserves distances.

In some cases, even an isometry may be too much to ask for and we can further weaken the requirement and just ask that the composition is a bi-Lipschitz map with bi-Lipschitz constant not too far from 1 (this is what we would mean by capturing the geometry), or that the composition is just a homeomorphism, i.e. a continuous map with a continuous inverse (what we would mean by capturing the topology). These requirements express in a weaker and weaker fashion that points that are close in data space should end up close in latent space and vice versa. The last version of the requirement closely corresponds to homeomorphic auto-encoding [10, 11].

We narrow down the question *is the interpretation of the latent space of a VAE warranted?* to *can a VAE capture topological and geometrical properties of latent space*?

For the VAE to have any chance of doing so, the latent space needs to be homeomorphic to the original space, otherwise one runs into the phenomenon of manifold mismatch. Therefore, to follow this route and investigate whether Variational Autoencoders are capable of capturing geometry or topology of the true latent variables, we needed to construct VAEs with manifolds as latent space, instead of just a linear vector space.

For special types of manifolds, such VAEs have already been developed: the hyperspherical VAEs allow for spheres in latent space [12], and Falorsi et al. implemented VAEs where the latent space is a Lie group [11].

2 Related work

Our work originated out of the search for algorithms that find semantically meaningful latent factors of data. The use of VAEs and their extensions to this end has mostly taken place in the context of *disentanglement of latent factors* [3, 8, 13]. Examples of extensions that aim at disentangling latent factors are the β-VAE [3], the factor-VAE [4], the β-TCVAE [6] and the DIP-VAE [7].

However, the examples in the introduction already show that in some situations, the topological structure of the latent space makes it practically impossible to disentangle latent factors. The latent factors are inherently, topologically entangled: in the case of a 3d rotation of an object, one cannot assign globally linearly independent angles of rotation.

Still, it is exactly global topological properties that we feel a VAE has a chance of capturing. What do we mean by this? One instance of 'capturing' topological structure is when the encoder and decoder of the VAE provide bijective, continuous maps between data and latent space, also called homeomorphic auto-encoding [11, 10]. This can only be done when the latent space has a particular topological structure, for instance that of a particular manifold.

We can also ask for more, that besides topological structure also geometric structure is captured. In that case, we require that distances in latent space carry some important meaning, for instance that distances in latent space are close to distances in data space, or to distances between ground-truth

latent variables in case they are known. Tosi et al. [14] and Arvanitidis et al. [15] take a related, but different point of view. They do not consider a standard metric or predetermined metric on latent space, but rather determine a Riemannian (pullback) metric that by construction reflects the distances in data space.

One of the main challenges when implementing a manifold as a latent space is the design of the reparametrization trick. In [12], a VAE was implemented with a hyperspherical latent space. To our understanding, they implemented a reparametrization function which was discontinuous.

If a manifold has the additional structure of a Lie group, this structure allows for a more straightforward implementation of the reparametrization trick [11]. In our work, we do not assume the additional structure of a Lie group, but develop a reparametrization trick that works for general submanifolds of Euclidean space, and therefore by the Whitney (respectively Nash) embedding theorem, for general closed (Riemannian) manifolds.

The method that we use has similarities with the approach of Hamiltonian Variational Inference [16]. Moreover, the implementation of a manifold as a latent space can be seen as enabling a particular, informative, prior distribution. In that sense, our work relates to [17, 18]. The prior distribution we implement is degenerate, in that it is does not assign mass to points outside of the manifold.

There are also other ways to implement approximate Bayesian inference on Riemannian manifolds. For instance, Liu and Zhu adapted the Stein variational gradient method to enable training on a Riemannian manifold [19]. However, their proposed method is rather expensive computationally.

The family of approximate posteriors that we implement is a direct generalization of the standard choice for a Euclidean VAE. Indeed, the Gaussian distributions are solutions to the heat equations, i.e. they are transition kernels of Brownian motion. One may want to increase the flexibility of the family of approximate posterior distributions, for instance by applying normalizing flows [20, 21, 22].

3 The Diffusion Variational Autoencoder ΔVAE

Our aim was to develop a VAE that also works if the latent space is just a manifold, and does not have the additional structure of a Lie group. This has led to the development of the Diffusion Variational Autoencoder (ΔVAE) [23].

Two important ingredients in a Variational Autoencoder are the reparametrization trick and a quick computation of the loss. Using a random-walk approximation of the diffusion kernels, we can implement the reparametrization trick. As a replacement of the exact evaluation of the KL loss in a standard Variational Autoencoder, we introduce an approximation of the KL term relying on a parametrix expansion.

A VAE has generally the following ingredients:

- a prior probability distribution $\mathbb{P}_Z$ on a latent space Z,
- a family of encoder distributions $\mathbb{Q}_Z^\alpha$ on Z, parametrized by α in a parameter space $\mathcal{A}$,
- a family of decoder distributions $\mathbb{P}_X^\beta$ on data space X, parametrized by β in a
- an encoder neural network $\boldsymbol{\alpha}$ which maps from data space X to the parameter space $\mathcal{A}$,
- a decoder neural network $\boldsymbol{\beta}$ which maps from latent space Z to parameter space $\mathcal{B}$.

The neural network weights are optimized to minimize the negated evidence lower bound (ELBO)

$$\mathcal{L}(x) = -\mathbb{E}_{z\sim\mathbb{Q}_Z^{\boldsymbol{\alpha}(x)}}\left[\log p_X^{\boldsymbol{\beta}(z)}(x)\right] + D_{\mathrm{KL}}\left(\mathbb{Q}_Z^{\boldsymbol{\alpha}(x)}||\mathbb{P}_Z\right).$$

The first term is called *reconstruction error* (RE); up to additive and multiplicative constants it equals the mean squared error (MSE). The second term is called the KL-loss.

In a very common implementation, both latent space Z and data space X are Euclidean, and the families of decoder and encoder distributions are multivariate Gaussian. The encoder and decoder networks then assign to a datapoint or a latent variable a mean and a variance respectively.

When we implement Z as a Riemannian manifold, we need to find an appropriate prior distribution, for which we will choose the normalized Riemannian volume measure, a family of encoder distributions $\mathbb{Q}_Z^\alpha$, for which we will take transition kernels of Brownian motion, and an encoder network mapping to the correct parameters.

Brownian motion on a Riemannian manifold We will briefly discuss Brownian motion on a Riemannian manifold, recommending lecture notes by Hsu [24] as a more extensive introduction. In the paper, we always assume that Z is a smooth Riemannian submanifold of Euclidean space, which is closed, i.e. it is compact and has no boundary. There are many different, equivalent definitions of Brownian motion. We present here the definition that is closest to our eventual approximation and implementation.

We will construct Brownian motion out of random walks on a manifold. We first fix a small time step $\tau > 0$. We will imagine a particle, jumping from point to point on the manifold after each time step, see also Fig. 2. It will start off at a point $z \in Z$. We describe the first jump, after which the process just repeats. After time τ, the particle makes a random jump $\sqrt{\tau}\epsilon_1$ from its current position, into the surrounding space, where ϵ_1 is distributed according to a radially symmetric distribution in $\mathbb{R}^n$ with identity covariance matrix. The position of the particle after the jump, $z + \sqrt{\tau}\epsilon_1$, will therefore in general not be on the manifold, so we project the particle back: The particle's new position will be $z_1 = P(z + \sqrt{\tau}\epsilon_1)$ where the closest-point-projection $P : \mathbb{R}^n \to Z$ assigns to every point $x \in \mathbb{R}^n$ the point in Z that is closest to x. After another time $\tau > 0$ the particle makes a new, independent, jump ϵ_2 according to the same radially symmetric distribution, and its new position will be $z_2 = P(P(z + \sqrt{\tau}\epsilon_1) + \sqrt{\tau}\epsilon_2)$. This process just repeats.

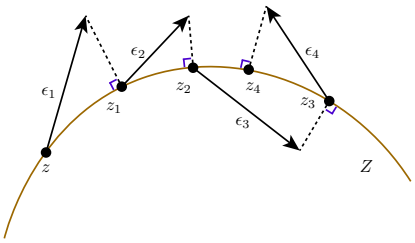


Figure 2: Random walk on a (one-dimensional) submanifold Z of $\mathbb{R}^2$, with time step $\tau = 1$.

Key to this construction, and also to our implementation, is the projection map P. It has nice properties, that follow from general theory of smooth manifolds. In particular, $P(x)$ smoothly depends on x, as long as x is not too far away from Z.

This way, for $\tau > 0$ fixed, we have constructed a random walk, a random path on the manifold. We can think of this path as a discretized version of Brownian motion. Let now τ_N be a sequence converging to 0 as $N \to \infty$. For fixed $N \in \mathbb{N}$, we can construct a random walk with time step τ_N, and get a random path $W^N : [0, \infty) \to Z$.

The random paths W^N converge as $N \to \infty$ to a random path W (in distribution). This random path W is called Brownian motion. The convergence statement can be made precise by for instance combining powerful, general results by [25] with standard facts from Riemannian geometry. But, because Riemannian manifolds are locally, i.e. when you zoom in far enough, very similar to Euclidean space, the convergence result essentially comes down to the central limit theorem and its upgraded version, Donsker's invariance theorem.

In fact, W can be interpreted as a Markov process, and even as a diffusion process. If A is a subset of Z, the probability that the Brownian motion $W(t)$ started at z is in the set A at time t is measured by a probability measure $\mathbb{Q}_Z^{t,z}$ applied to the set A. We denote the density of this measure with respect to the standard Riemannian volume measure Vol by $q_Z(t; z, \cdot)$. The function q_Z is sometimes referred to as the heat kernel.

Let us close this subsection with an alternative description of the function q_Z. It is also characterized by the fact that for every function $u_0 : Z \to \mathbb{R}$, the solution to the partial differential equation

$$\begin{cases} \partial_t u = \frac{1}{2}\Delta u & \text{on } (0, \infty) \times Z \\ u(t = 0, \cdot) = u_0 & \text{on } Z \end{cases}$$

is given by

$$u(t, z) = \int_Z u_0(y) q_Z(t; z, y) dy.$$

Riemannian manifold as latent space A ΔVAE is a VAE with a Riemannian submanifold of Euclidean space as a latent space, and the transition probability measures of Brownian motion $\mathbb{Q}_Z^{t,z}$ as a parametric family of encoder distributions. We propose the uniform distribution for $\mathbb{P}_Z$, which is the normalized standard measure on a Riemannian manifold (although the choice of prior distribution could easily be generalized).

As in the standard VAE, we then implement functions $\mathbf{z} : X \to Z$ and $\mathbf{t} : X \to (0, \infty)$ as neural networks.

We optimize the weights in the network, aiming to minimize the average loss for the loss function

$$-\mathbb{E}_{z\sim\mathbb{Q}_Z^{\mathbf{t}(x),\mathbf{z}(x)}}\left[\log p_X^{\boldsymbol{\beta}(z)}(x)\right] + D_{\mathrm{KL}}\left(\mathbb{Q}_Z^{\mathbf{t}(x),\mathbf{z}(x)} \| \mathbb{P}_Z\right).$$

The first integral can often only be approached by sampling, and in that case it is often advantageous to perform a change of variables, commonly known as the *reparametrization trick* [1].

Approximate reparametrization by random walk Instead, we construct an *approximate* reparametrization map by approximating Brownian motion by a random walk, similar to how we defined it in this paper. Starting from a point z on the manifold, we set a random step in *ambient space* $\mathbb{R}^n$. We then project back to the manifold and repeat: we take a new step and project back to the manifold. In total, we take N steps, see Fig. 2.

We define the function $g : \mathcal{E}^N \times (0, \infty) \times Z \to Z$ by

$$g(\epsilon_1, \ldots, \epsilon_N, t, z) = P\left(\cdots P\left(P\left(z + \sqrt{\tfrac{t}{N}}\epsilon_1\right) + \sqrt{\tfrac{t}{N}}\epsilon_2\right)\cdots + \sqrt{\tfrac{t}{N}}\epsilon_N\right).$$

If we take $\epsilon_1, \ldots, \epsilon_N$ as i.i.d. random variables, distributed according to a radially symmetric distribution, then $y = g(\epsilon_1, \ldots, \epsilon_N, z)$ is approximately distributed as a random variable with density $q_Z(t; z, \cdot)$. The computational complexity of the sampling is linear in N. Yet the approximation is very accurate for small times t, even for small values of N, if we take $\epsilon_1, \ldots, \epsilon_N$ approximately Gaussian. We have set $N = 10$ throughout the presented results. The observation that for small times, the diffusion kernel q_Z is approximately Gaussian, is also very helpful in approximating the KL term in the loss.

Approximation of the KL-divergence Unlike the standard VAE, or the hyperspherical VAE with the Von-Mises distribution, the KL-term cannot be computed exactly for the ΔVAE. There are several techniques one could use to get, nonetheless, a good approximation of the KL divergence. We have implemented an asymptotic approximation, which we will describe first.

Asymptotic approximation We can use short-term asymptotics, i.e. a parametrix expansion, of the heat kernel on Riemannian manifolds to obtain asymptotic expansions of the entropy.

Proposition 1 *The KL divergence follows the following formal asymptotic expansion, where d is the dimensionality of Z, and Sc is the scalar curvature of the manifold Z in z.*

$$\begin{aligned} D_{\mathrm{KL}}(\mathbb{Q}^{t,z} \| \mathbb{P}_Z) &= \int_Z q_Z(t; z, y) \log q_Z(t; z, y) dy + \log \mathrm{Vol}(Z) \\ &= -\frac{d}{2}\log(2\pi t) - \frac{d}{2} + \log \mathrm{Vol}(Z) + \frac{1}{4}\mathsf{Sc}\, t + o(t). \end{aligned}$$

We give the derivation of the expansion in Appendix A. In our implementation, we restrict t so that it cannot become too large, thus ensuring a certain accuracy of the asymptotic expansion.

Besides the asymptotic approximation, one may also choose to approximate the heat kernel numerically or use Monte Carlo approximation.

4 Experiments

4.1 Periodic translation of pictures

As a test of the ΔVAE we trained a ΔVAE with a flat torus as a latent space on a dataset consisting of periodically translated versions of the same picture. That is, we took a fixed picture, sampled many random elements from the flat torus (depicted in Fig. 3a) and shifted the picture over those elements. The ΔVAE only gets to see the dataset of resulting pictures, but not the corresponding shifts.

After training, the encoder of the ΔVAE maps pictures to latent space, and the important question is whether it arranges the picture according to the latent shifts. To judge that, we have sampled a regular

grid of shifted pictures, and plotted their image under the encoder in latent space in Fig. 3c, color coded according to the scheme in Fig. 3a. The resulting picture is up to a shift almost the original one.

4.2 Rotations of objects

We have also investigated the capabilities of the ΔVAE in capturing the underlying topological structure for a synthetic dataset consisting of rendered images from a 3D model of an airplane within the ModelNet [26] data set. The images consists of gray-scale renders of 64×64 pixels, showing of the 3D model centered in a frame of reference and rotated around the z-axis. The angle of rotation for each image is chosen from a regular partition of the interval S^1.

Fig. 1 shows the latent variables of a ΔVAE trained with S^1 as a latent space and 100 rendered images. The color encoding represents the true angles at which each of the images was generated. The ΔVAE is capable of capturing the topological of the latent orientation.

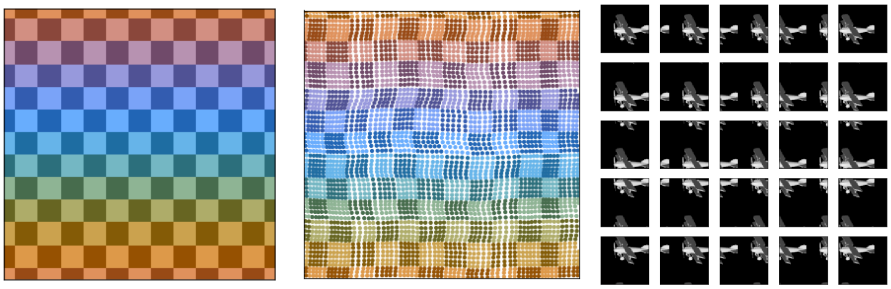
As a small discussion, we should mention that the topological structure is not captured in every training run, but depends on the random initialization of the Variational Autoencoder.

4.3 ΔVAEs for MNIST

We then trained ΔVAEs on binarized MNIST [27]. We show the manifolds as latent space with encoded MNIST digits in Fig. 4. When MNIST is trained on different latent spaces, different adjacency structures between digits may become apparent, providing topological information.

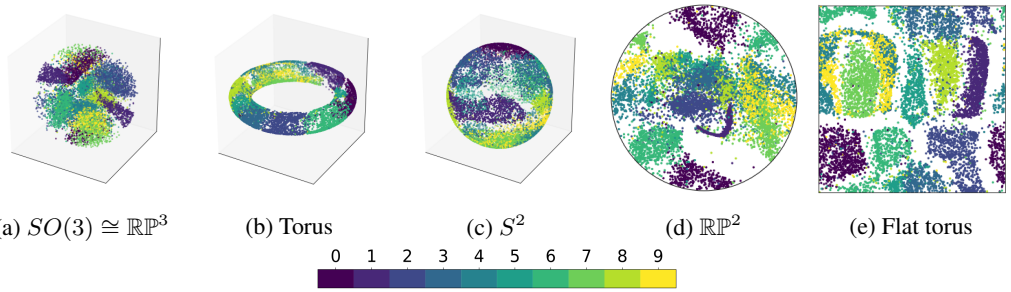
The $SO(3)$ is isometric to a scaling of the $\mathbb{RP}^3$ (with natural choices of Riemannian metrics). Although we have implemented an embedding and projection map for this embedding for the $SO(3)$ directly based on an SVD decomposition to find the nearest orthogonal matrix, training on the $\mathbb{RP}^3$ with the following trick was faster and we only present these results.

For training the projective spaces, we used an additional trick, where instead of embedding $\mathbb{RP}^d$ in a Euclidean space, we embed S^d in $\mathbb{R}^{d+1}$, and make the decoder neural network *even* by construction (i.e. the decoder applied to a point s on the sphere equals the decoder applied to a point $-s$). Then, an encoder and decoder to and from the $\mathbb{RP}^3$ are defined implicitly. However, it must be noted that this setup does not allow for a homeomorphic encoding (because $\mathbb{RP}^d$ does not embed in S^d).



(a) Color scheme (b) Latent space flat torus (c) Reconstructed images

Figure 3: (a) Color scheme with periodic boundaries used to identify the horizontal and vertical translation that generated the training data. Results shown correspond to the training of a ΔVAE with a flat torus as latent space on a rendered image of an airplane subject to periodical translation along its height and width. Figures show the resulting embeddings in the latent space manifold with periodic boundaries using color scheme for translations (b) and the reconstructed images from a regular grid over the latent variables (c).



(a) $SO(3) \cong \mathbb{RP}^3$ (b) Torus (c) S^2 (d) $\mathbb{RP}^2$ (e) Flat torus

Figure 4: Latent space representation of MNIST for several manifolds. The projective spaces are represented by a 3- and 2-dimensional ball respectively, for which every point on the boundary is identified with its reflection through the center. The effect of this identification can be seen, since the same digits that map close to a point on the boundary also map close to the reflected point.

The numerically computed ELBO, reconstruction error and KL-divergence are shown in Table 1 together with the estimated log-likelihood for a test dataset of the binarized MNIST. We provide a comparison with the values obtained in [12] trained on a spherical latent space S^2 with a uniform prior. Additionally we present the results obtained in [28] trained on a latent space consisting of two circular independent latent variables with a uniform prior, which can be directly compared to the ΔVAE with a flat torus latent space.

The ΔVAE achieves similar log-likelihood estimates with respect to the results on $\mathcal{S}^2$ from [12]. On the other hand, the results for the ΔVAE trained on a flat torus have a lower log-likelihood compared to the results from [28] (higher values are better)

Table 1: Numerical results for ΔVAEs trained on binarized MNIST. The values indicate mean and standard deviation over 10 runs. The columns represent the (data-averaged) log-likelihood estimate (LL), Evidence Lower Bound (ELBO), KL-divergence (KL) and reconstruction error (RE) evaluated on the test data. For comparison we present results for S^2 as reported by [12] and for the flat torus as reported by [28].

MANIFOLD	LL	ELBO	KL	RE
S^2	-132.20±0.39	-134.67±0.47	7.23±0.05	-127.44±0.47
EMBEDDED TORUS	-132.79±0.53	-137.37±0.59	9.14±0.18	-128.23±0.67
FLAT TORUS	-131.73±0.69	-139.97±0.78	12.91±0.08	-127.07±0.81
$\mathbb{RP}^3$	-125.27±0.37	-128.17±0.58	9.38±0.12	-118.79±0.60
$\mathbb{RP}^2$	-135.87±0.66	-138.13±0.72	7.02±0.12	-131.11±0.73
$\mathbb{R}^3$	-124.71±0.93	-128.01±1.05	9.12±0.09	-118.89±1.01
$\mathbb{R}^2$	-134.17±0.53	-136.61±0.64	7.05±0.06	-129.56±0.63
S^2 [12]	-132.50±0.83	-133.72±0.85	7.28±0.14	-126.43±0.91
FLAT TORUS[28]	-127.60±0.40	-	-	-

Estimation of log-likelihood For the evaluation of the proposed methods we have estimated the log-likelihood of the test dataset according to the importance sampling presented in [29]. The approximate log-likelihood of datapoint x is calculated by sampling M latent variables $z^{(1)}, \ldots, z^{(M)}$ according to the approximate posterior $\mathbb{Q}_Z^{\mathbf{t}(x),\mathbf{z}(x)}$. The estimated log-likelihood for datapoint x is given by

$$\log p_X^{\boldsymbol{\beta}}(x) \approx \log \left(\frac{1}{M} \sum_{m=1}^{M} \frac{p_X^{\boldsymbol{\beta}(z^{(m)})}(x) p_Z(z^{(m)})}{q_Z(\mathbf{t}(x); \mathbf{z}(x), z^{(m)})} \right).$$

The log-likelihood estimates presented in Table 1 are obtained with $M = 1000$ samples for each datapoint, averaged over all datapoints.

5 Conclusion

Our motivation to develop Diffusion Variational Autoencoders, was to investigate to which extent VAEs find semantically meaningful latent variables, and more specifically, whether they can capture topological and geometrical structure in datasets. By allowing for an arbitrary manifold as a latent space, ΔVAEs can remove obstructions to capturing such structure.

Our experiments with translations of periodic images and rotations of objects show that a simple implementation of a ΔVAE with a flat torus as latent space is capable of capturing topological properties, although depending on the random initialization it does not always succeed.

Acknowledgment

This work has received funding from the Electronic Component Systems for European Leadership Joint Undertaking under grant agreement No 737459 (project Productive 4.0).

References

[1] Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. In *ICLR*, 2014.

[2] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint*, arXiv:1401.4082, 2014.

[3] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. β-vae: Learning basic visual concepts with a constrained variational framework. *ICLR*, 2017.

[4] Hyunjik Kim and Andriy Mnih. Disentangling by factorising. *arXiv preprint*, arXiv:1802.05983, 2018.

[5] Jack Klys, Jake Snell, and Richard Zemel. Learning latent subspaces in variational autoencoders. In *Advances in Nueral Information Processing Systems 31*, 2018.

[6] Tian Qi Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. Isolating sources of disentanglement in Variational Autoencoders. In *Advances in Neural Information Processing Systems 31*. 2018.

[7] Abhishek Kumar, Prasanna Sattigeri, and Avinash Balakrishnan. Variational inference of distentangled latent concepts from unlabeled observations. *ICLR*, 2018.

[8] Irina Higgins, David Amos, David Pfau, Sebastien Racaniere, Loic Matthey, Danilo Rezende, and Alexander Lerchner. Towards a definition of disentangled representations. *arXiv preprint*, arXiv:1812.02230, 2018.

[9] Luis A. Pérez Rey. Latent variable separation with Variational Autoencoders. *Master's thesis, https://pure.tue.nl/ws/portalfiles/portal/125630525/Final_Version_Luis_IAM_309.pdf.*

[10] Pim de Haan and Luca Falorsi. Topological constraints on homeomorphic auto-encoding. *arXiv preprint*, arXiv:1812.10783, 2018.

[11] Luca Falorsi, Pim de Haan, Tim R. Davidson, Nicola De Cao, Maurice Weiler, Patrick Forré, and Taco S. Cohen. Explorations in homeomorphic variational auto-encoding. *arXiv preprint*, arXiv:1807.04689, 2018.

[12] Tim R. Davidson, Luca Falorsi, Nicola De Cao, Thomas Kipf, and Jakub M. Tomczak. Hyperspherical variational auto-encoders. In *34th Conference on Uncertainty in Artificial Intelligence.* 2018.

[13] Christopher P. Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in β-VAE. *arXiv preprint*, arXiv:1804.03599, 2018.

[14] Alessandra Tosi, Sören Hauberg, Alfredo Vellido, and Neil D. Lawrence. Metrics for probabilistic geometries. In *UAI 2014*, 2014.

[15] Georgios Arvanitidis, Lars Kai Hansen, and Søren Hauberg. Latent space oddity: on the curvature of deep generative models. In *ICLR 2018*, 2018.

[16] Tim Salimans, Diederik Kingma, and Max Welling. Markov Chain Monte Carlo and Variational Inference: Bridging the gap. In *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

[17] Nat Dilokthanakul, Pedro A. M. Mediano, Marta Garnelo, Matthew C. H. Lee, Hugh Salimbeni, Kai Arulkumaran, and Murray Shanahan. Deep unsupervised clustering with Gaussian mixture Variational Autoencoders. *arXiv preprint*, arXiv:1611.02648, 2016.

[18] Jakub M. Tomczak and Max Welling. VAE with a VampPrior. *arXiv preprint*, arXiv:1705.07120, 2017.

[19] Chang Liu and Jun Zhu. Riemannian Stein variational gradient descent for Bayesian inference. *arXiv preprint*, arXiv:1711.11216, 2017.

[20] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

[21] Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved Variational Inference with Inverse Autoregressive Flow. In *Advances in Neural Information Processing Systems 29*. 2016.

[22] Mevlana C. Gemici, Danilo Rezende, and Shakir Mohamed. Normalizing flows on Riemannian manifolds. *arXiv preprint*, arXiv:1611.02304, 2016.

[23] Luis A. Pérez Rey, Vlado Menkovski, and Jacobus W. Portegies. Diffusion variational autoencoders. *arXiv preprint*, arXiv:1901.08991.

[24] Elton P Hsu. *A brief introduction to Brownian motion on a Riemannian manifold*. Lecture notes, 2008.

[25] Erik Jørgensen. The central limit problem for geodesic random walks. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 32(1-2):1–64, 1975.

[26] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3D ShapeNets: A Deep Representation for Volumetric Shapes. 2014.

[27] Ruslan Salakhutdinov and Iain Murray. On the quantitative analysis of deep belief networks. In *Proceedings of the 25th International Conference on Machine Learning*, 2008.

[28] Michael Figurnov, Shakir Mohamed, and Andriy Mnih. Implicit Reparameterization Gradients. *Advances in Neural Information Processing Systems 31*, 2018.

[29] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. *ICLR*, 2016.

[30] Marcel Berger, Paul Gauduchon, and Edmond Mazet. Le spectre d'une variété riemannienne. In *Le Spectre d'une Variété Riemannienne*, pages 141–241. Springer, 1971.

A Asymptotic expansion KL divergence

In this appendix, we derive a short-term asymptotic expansion of the KL divergence term for arbitrary Riemannian manifolds presented in Section 3 and given by

$$\begin{aligned} D_{\mathrm{KL}}(\mathbb{Q}^{t,z}\|\mathbb{P}_Z) &= \int_Z q_Z(t;z,y)\log q_Z(t;z,y)dy + \log \mathrm{Vol}(Z) \\ &= -\frac{d}{2}\log(2\pi t) - \frac{d}{2} + \log \mathrm{Vol}(Z) + \frac{1}{4}\mathsf{Sc}\, t + o(t). \end{aligned}$$

We will focus on the derivation of the integral in the KL divergence

$$\int_Z q_Z(t;z,w)\log q_Z(t;z,w)dw. \tag{1}$$

We base the expansion on a short-term expansion of the heat kernel itself, also known as a parametrix expansion, cf. [30]

$$\begin{aligned} q_Z(t;z,w) &:= \frac{1}{(2\pi t)^{d/2}} \exp\left(-\frac{r^2}{2t}\right)(u_0(z,w) + tu_1(z,w) + o(t)) \\ &= \frac{1}{(2\pi t)^{d/2}} \exp\left(-\frac{r^2}{2t}\right) u_0(z,w)\left(1 + t\frac{u_1(z,w)}{u_0(z,w)} + o(t)\right), \end{aligned} \tag{2}$$

where r is the distance between z and w, and where we use the notation $o(t)$ for terms that go to zero faster than t.

Because the heat kernel decays exponentially, for calculating the asymptotic behavior in (1), only the behavior of the function $u_j(z,w)$ for z close to w is relevant. We choose normal coordinates y^i centered at z (so z corresponds to $y^i = 0$, and $r^2 = |y|^2$), and Taylor expand the functions u_j in terms of y^i. It is helpful to keep in mind as a rule of thumb, that a monomial of degree k in y^i corresponds to a factor of degree $k/2$ in t in the final asymptotic expansion of the integral (1).

We split the logarithm of q_Z in four terms,

$$\log q_Z(t;z,w) = J_1(t;z,w) + J_2(t;z,w) + J_3(t;z,w) + J_4(t;z,w) \tag{3}$$

where

$$\begin{aligned} J_1(t;z,w) &:= -\frac{d}{2}\log(2\pi t) \\ J_2(t;z,w) &:= -\frac{r^2}{2t} \\ J_3(t;z,w) &:= \log u_0(z,w) \\ J_4(t;z,w) &:= \log\left(1 + t\frac{u_1(z,w)}{u_0(z,w)} + o(t)\right) \\ &= t\frac{u_1(z,w)}{u_0(z,w)} + o(t) \end{aligned}$$

where we used the Taylor expansion of the logarithm to get the second line of J_4.

Write

$$\theta(z,w) = \sqrt{\det(g_{ij}(w))}$$

where g_{ij} are the coefficients of the metric in the normal coordinates y^i centered at z.

In the parametrix expansion, the function u_0 equals

$$u_0(z,w) = \frac{1}{\sqrt{\theta(z,w)}}$$

see p. 208 of [30].

Key are the following asymptotic expansions in normal coordinates y^i centered at z,

$$\theta(0,y) = 1 - \frac{1}{6}\mathrm{Ric}_{ij}y^i y^j + O(|y|^3),$$

and

$$\sqrt{\theta(0,y)} = 1 - \frac{1}{12}\mathrm{Ric}_{ij}y^i y^j + O(|y|^3). \tag{4}$$

As a consequence, we have the following asymptotic expansion for u_0

$$u_0(0,y) = \frac{1}{\sqrt{\theta(0,y)}} = 1 + \frac{1}{12}\mathrm{Ric}_{ij}y^i y^j + O(|y|^3). \tag{5}$$

Next, we use that the function u_1 is given by the following integral (see (E.III.1) in [30], but note that they have a different sign convention for the Laplacian, see formula (G.III.2) in their book, and that we use the stochastic normalization in the heat equation, which accounts for an extra factor of $\frac{1}{2}$)

$$u_1(z,w) = \frac{1}{2}\theta^{-1/2}(z,w)\int_0^1 \theta^{1/2}\Big(z, \exp_z\left(\tau \exp_z^{-1}(w)\right)\Big)\Delta u_0\Big(z, \exp_z\left(\tau \exp_z^{-1}(w)\right)\Big)d\tau$$

where the Laplacian is taken in the second argument and where $\exp_z$ denotes the exponential map based at z.

Because the fraction $u_1(z,w)/u_0(z,w)$ gets multiplied by a factor t in the parametrix expansion (2), we will later only need the zeroth order term of $u_1(z,w)/u_0(z,w)$ and $u_1(z,w)$. Since

$$\Delta u_0(0,y) = \frac{1}{12}2\mathrm{tr}(\mathrm{Ric}) + O(|y|) = \frac{1}{6}\mathsf{Sc} + O(|y|),$$

we get from the integral formula that

$$u_1(0,y) = \frac{1}{12}\mathsf{Sc} + O(|y|)$$

and, by using (5), that

$$\frac{u_1(0,y)}{u_0(0,y)} = \frac{1}{12}\mathsf{Sc} + O(|y|). \tag{6}$$

To compute the integral

$$\int_Z q_Z(t;z,w)\log q_Z(t;z,w)dw$$

we split the logarithm in four terms J_i as in (3). The first term gives

$$\begin{aligned}\int_Z q_Z(t;z,w)J_1(t;z,w)dw &= -\frac{d}{2}\int_Z q_Z(t;z,w)\log(2\pi t)\,dw \\ &= -\frac{d}{2}\log(2\pi t).\end{aligned}$$

The fourth term is also easy and gives

$$\begin{aligned}\int_Z q_Z(t;z,w)J_4(t;z,w)dw &= \int_Z q_Z(t;z,w)\frac{1}{12}\mathsf{Sc}\,t\,dw + o(t) \\ &= \frac{1}{12}\mathsf{Sc}\,t + o(t).\end{aligned}$$

Now let us look at

$$\begin{aligned}&\int_Z q_Z(t;z,w)J_2(t;z,w)dw = -\int_Z q_Z(t;z,w)\frac{r^2}{2t}dw \\ &= -\int_{\mathbb{R}^d}\frac{1}{(2\pi t)^{d/2}}\exp\left(-\frac{|y|^2}{2t}\right)\frac{|y|^2}{2t}\frac{1}{\sqrt{\theta(0,y)}}\left(1+t\frac{u_1(0,y)}{u_0(0,y)}+o(t)\right)\theta(0,y)dy \\ &= -\int_{\mathbb{R}^d}\frac{1}{(2\pi t)^{d/2}}\exp\left(-\frac{|y|^2}{2t}\right)\frac{|y|^2}{2t}\left(1+t\frac{u_1(0,y)}{u_0(0,y)}+o(t)\right)\sqrt{\theta(0,y)}dy.\end{aligned}$$

We substitute the asymptotic behavior of $u_1(z,w)/u_0(z,w)$ from (6) and the asymptotic behavior of $\sqrt{\theta(0,y)}$ from (4),

$$\begin{aligned}&\left(1+t\frac{u_1(0,y)}{u_0(0,y)}+o(t)\right)\sqrt{\theta(0,y)} \\ &= \left(1+t\left(\frac{1}{12}\mathsf{Sc}+O(|y|)\right)+o(t)\right)\left(1-\frac{1}{12}\mathrm{Ric}_{ij}y^iy^j+O(|y|^3)\right).\end{aligned}$$

We expand the factors and integrate. Note that the integral of an $O(|y|^k)$ term against the Gaussian measure can be, after integration, estimated by a term of $O(t^{k/2})$. We therefore find

$$\begin{aligned}&\int_Z q_Z(t;z,w)J_2(t;z,w)dw \\ &= -\int_{\mathbb{R}^d}\frac{1}{(2\pi t)^{d/2}}\exp\left(-\frac{|y|^2}{2t}\right)\frac{|y|^2}{2t}\left(1+\frac{1}{12}\mathsf{Sc}\,t-\frac{1}{12}\mathrm{Ric}_{ij}\,y^iy^j\right)dy+o(t).\end{aligned}$$

All that is left to do is compute Gaussian integrals, which follow from the moments of a multidimensional Gaussian distribution with mean zero and a covariance of t times the identity matrix. In particular, fixing i, we have

$$\begin{aligned}
&\int_{\mathbb{R}^d} \frac{1}{(2\pi t)^{d/2}} \exp\left(-\frac{|y|^2}{2t}\right) |y|^2 (y^i)^2 dy \\
&= \int_{\mathbb{R}} \frac{1}{\sqrt{2\pi t}} \exp\left(-\frac{(y^i)^2}{2t}\right) (y^i)^4 dy \\
&\quad + \sum_{j \neq i} \left(\int_{\mathbb{R}} \frac{1}{\sqrt{2\pi t}} \exp\left(-\frac{(y^j)^2}{2t}\right) (y^j)^2 dy^j\right) \left(\int_{\mathbb{R}} \frac{1}{\sqrt{2\pi t}} \exp\left(-\frac{(y^i)^2}{2t}\right) (y^i)^2 dy^i\right) \\
&= 3t^2 + (d-1)t^2.
\end{aligned}$$

We find

$$\begin{aligned}
&\int_Z q_Z(t;z,w) J_2(t;z,w) dw \\
&= -\frac{d}{2} - \frac{d}{24} \mathsf{Sc}\, t + \frac{1}{24} t(3 + (d-1)) \sum_{i=1}^{d} \mathrm{Ric}_{ii} \\
&= -\frac{d}{2} - \frac{d}{24} \mathsf{Sc}\, t + \frac{1}{24} \mathsf{Sc}\, t(d+2) \\
&= -\frac{d}{2} + \frac{1}{12} \mathsf{Sc}\, t.
\end{aligned}$$

Finally, we consider

$$\begin{aligned}
\int_Z q_Z(t;z,w) J_3(t;z,w) dw &= \int_Z q_Z(t;z,w) \log u_0(z,w) dw \\
&= \int_{\mathbb{R}^d} \frac{1}{(2\pi t)^{d/2}} \exp\left(-\frac{|y|^2}{2t}\right) \frac{1}{12} \mathrm{Ric}_{ij} y^i y^j dy + o(t) \\
&= \frac{1}{12} \mathsf{Sc}\, t + o(t).
\end{aligned}$$

If we add all contributions, we obtain

$$\int_Z q_Z(t;z,w) \log q_z(t;z,w) dw = -\frac{d}{2} \log(2\pi t) - \frac{d}{2} + \frac{1}{4} \mathsf{Sc}\, t + o(t).$$