Journal of Machine Learning Research 25 (2024) 1-35 Submitted 3/23; Revised 2/24; Published 6/24

Manifold Learning by Mixture Models of VAEs for Inverse Problems

Giovanni S. Alberti GIOVANNI.ALBERTI@UNIGE.IT
MaLGa Center
Department of Mathematics, Department of Excellence 2023–2027
University of Genoa, Italy

Johannes Hertrich J.HERTRICH@UCL.AC.UK
Department of Computer Science
University College London,
London, United Kingdom

Matteo Santacesaria MATTEO.SANTACESARIA@UNIGE.IT
MaLGa Center
Department of Mathematics, Department of Excellence 2023–2027
University of Genoa, Italy

Silvia Sciutto SILVIA.SCIUTTO@EDU.UNIGE.IT
MaLGa Center
Department of Mathematics, Department of Excellence 2023–2027
University of Genoa, Italy

Editor: Amos Storkey

Abstract

Representing a manifold of very high-dimensional data with generative models has been shown to be computationally efficient in practice. However, this requires that the data manifold admits a global parameterization. In order to represent manifolds of arbitrary topology, we propose to learn a mixture model of variational autoencoders. Here, every encoder-decoder pair represents one chart of a manifold. We propose a loss function for maximum likelihood estimation of the model weights and choose an architecture that provides us the analytical expression of the charts and of their inverses. Once the manifold is learned, we use it for solving inverse problems by minimizing a data fidelity term restricted to the learned manifold. To solve the arising minimization problem we propose a Riemannian gradient descent algorithm on the learned manifold. We demonstrate the performance of our method for low-dimensional toy examples as well as for deblurring and electrical impedance tomography on certain image manifolds.

Keywords: manifold learning, mixture models, variational autoencoders, Riemannian optimization, inverse problems

1. Introduction

Manifold learning. The treatment of high-dimensional data is often computationally costly and numerically unstable. Therefore, in many applications, it is important to find a low-dimensional representation of high-dimensional data sets. Classical methods, like the principal component analysis (PCA, Pearson, 1901), assume that the data is contained in a

low-dimensional subspace. However, for complex data sets this assumption appears to be too restrictive, particularly when working with image data sets. Therefore, recent methods rely on the so-called manifold hypothesis (Bengio et al., 2013), stating that even complex and high-dimensional data sets are contained in a low-dimensional manifold. Based on this hypothesis, in recent years, many successful approaches have been based on generative models, able to represent high dimensional data in $\mathbb{R}^n$ by a generator $D\colon \mathbb{R}^d \to \mathbb{R}^n$ with $d \ll n$: these include generative adversarial networks (GANs, Goodfellow et al., 2014), variational autoencoders (VAEs, Kingma and Welling, 2014), injective flows (Kothari et al., 2021) and score-based diffusion models (Song and Ermon, 2019; Ho et al., 2020). For a survey on older approaches to manifold learning, the reader is referred to Ma and Fu (2011); Izenman (2012) and to the references therein.

Learning manifolds with multiple charts. Under the assumption that D is injective, the set of generated points $\{D(z) : z \in \mathbb{R}^d\}$ forms a manifold that approximates the training set. However, this requires that the data manifold admits a global parameterization. In particular, it must not be disconnected or contain holes. In order to model disconnected manifolds, Falck et al. (2021); Jiang et al. (2017); Kivva et al. (2022); Pineau and Lelarge (2018) propose to model the latent space of a VAE by a Gaussian mixture model. This enables the authors to capture multimodal probability distributions. However, this approach struggles with modelling manifolds with holes since either the injectivity of the generator is violated or it is impossible to model overlapping charts. Similarly, Davidson et al. (2018); Mathieu et al. (2019); Rey et al. (2020) propose latent distributions defined on Riemannian manifolds for representing general topologies. Massa et al. (2022) embed the manifold into a higher-dimensional space, in the spirit of Whitney embedding theorem. However, these approaches have the drawback that the topology of the manifold has to be known a priori, which is usually not the case in practice.

Here, we focus on the representation of the data manifold by several charts. A chart provides a parameterization of an open subset from the manifold by defining a mapping from the manifold into a Euclidean space. Then, the manifold is represented by the collection of all of these charts, which is called atlas. For finding these charts, Cohn et al. (2021); Floryan and Graham (2022); Pitelis et al. (2013); Sidheekh et al. (2022) propose the use of clustering algorithms. By default, these methods do not provide an explicit formulation of the resulting charts. As a remedy, Brand (2002); Pitelis et al. (2013) use linear or kernelized embeddings. Sidheekh et al. (2022) propose to learn for each chart again a generative model. However, these approaches often require a large number of charts and are limited to relatively low data dimensions. The idea of representing the charts by generative models is further elaborated by Korman (2018, 2021); Schonsheck et al. (2019). Here, the authors proposes to train at the same time several (non-variational) autoencoders and a classification network that decides for each point to which chart it belongs. In contrast to the clustering-based algorithms, the computational effort scales well for large data dimensions. On the other hand, the numerical examples in the corresponding papers show that the approach already has difficulties to approximate small toy examples like a torus.

In this paper, we propose to approximate the data manifold by a mixture model of VAEs. Using Bayes theorem and the ELBO approximation of the likelihood term we derive a loss function for maximum likelihood estimation of the model weights. Mixture models of

generative models for modelling disconnected data sets were already considered by Banijamali et al. (2017); Hoang et al. (2018); Locatello et al. (2018); Stolberg-Larsen and Sommer (2022); Ye and Bors (2022). However, they are trained in a different way and to the best of our knowledge none of those is used for manifold learning.

Inverse Problems on Manifolds. Many problems in applied mathematics and image processing can be formulated as inverse problems. Here, we consider an observation y which is generated by

$$y = \mathcal{G}(x) + \eta,$$

where $\mathcal{G}\colon \mathbb{R}^n \to \mathbb{R}^m$ is an ill-posed or ill-conditioned, possibly nonlinear forward operator and η represents additive noise. Reconstructing the input x directly from the observation y is usually not possible due to the ill-posed operator and the high dimension of the problem. As a remedy, the incorporation of prior knowledge is required. This is usually achieved by using regularization theory, namely, by minimizing the sum of a data fidelity term $F(x)$ and a regularizer $R(x)$, where F describes the fit of x to y and R incorporates the prior knowledge. With the success of deep learning, data-driven regularizers became popular (Altekrüger et al., 2023; Arridge et al., 2019; Goujon et al., 2023; Hertrich et al., 2022; Lunz et al., 2018).

In this paper, we consider a regularizer which constraints the reconstruction x to a learned data manifold $\mathcal{M}$. More precisely, we consider the optimization problem

$$\hat{x} = \arg\min_x F(x) \quad \text{subject to } x \in \mathcal{M},$$

where $F(x) = \frac{1}{2}\|\mathcal{G}(x) - y\|^2$ is a data-fidelity term. This corresponds to the regularizer $R(x)$ which is zero for $x \in \mathcal{M}$ and infinity otherwise. When the manifold admits a global parameterization given by one single generator D, Alberti et al. (to appear); Chen et al. (2020); Duff et al. (2021); González et al. (2022) propose to reformulate the problem as $\hat{x} = \hat{z}$, where $\hat{z} = \arg\min_z F(D(x))$. Since this is an unconstrained problem, it can be solved by gradient based methods. However, since we consider manifolds represented by several charts, this reformulation cannot be applied. As a remedy, we propose to use a Riemannian gradient descent scheme. In particular, we derive the Riemannian gradient using the decoders and encoders of our manifold and propose two suitable retractions for applying a descent step into the gradient direction.

To emphasize the advantage of using multiple generators, we demonstrate the performance of our method on numerical examples. We first consider some two- and three-dimensional toy examples. Finally, we apply our method to deblurring and to electrical impedance tomography (EIT), a nonlinear inverse problem consisting in the reconstruction of the leading coefficient of a second order elliptic PDE from the knowledge of the boundary values of its solutions (Cheney et al., 1999). The code of the numerical examples is available online.[1]

Outline. The paper is organized as follows. In Section 2.1, we revisit VAEs and fix the corresponding notations. Afterwards, in Section 3, we introduce mixture models of VAEs

1. The code is available at `https://github.com/johertrich/Manifold_Mixture_VAEs`

for learning embedded manifolds of arbitrary dimensions and topologies. Here, we focus particularly on the derivation of the loss function and of the architecture, which allows us to access the charts and their inverses. For minimizing functions defined on the learned manifold, we propose a Riemannian gradient descent scheme in Section 4. We provide numerical toy examples for one and two dimensional manifolds in Section 5. In Section 6, we discuss the applications to deblurring and to electrical impedance tomography. Conclusions are drawn in Section 7.

2. Background on Variational Autoencoders and Manifolds

In this section, we revisit the technical background of the paper. First, we recall the concept of VAEs, their training procedure and of a learned latent space with normalizing flows. Afterwards, we give a short literature review on manifold learning with VAEs, and some basic notions from differential geometry.

2.1 Variational Autoencoders for Manifold Learning

In this paper, we assume that we are given data points $x_1, \dots, x_N \in \mathbb{R}^n$ for a large dimension n. In order to reduce the computational effort and to regularize inverse problems, we assume that these data-points are located in a lower-dimensional manifold. We aim to learn the underlying manifold from the data points $x_1, \dots, x_N$ with a VAE (Kingma and Welling, 2014, 2019).

A VAE aims to approximate the underlying high-dimensional probability distribution P_X of the random variable X with a lower-dimensional latent random variable $Z \sim P_Z$ on $\mathbb{R}^d$ with $d < n$, by using the data points $x_1, \dots, x_N$. To this end, we define a decoder $D\colon \mathbb{R}^d \to \mathbb{R}^n$ and an encoder $E\colon \mathbb{R}^n \to \mathbb{R}^d$. The decoder approximates P_X by the distribution $P_{\tilde{X}}$ of a random variable $\tilde{X} \coloneqq D(Z) + \eta$, where $\eta \sim \mathcal{N}(0, \sigma_x^2 I_n)$. Vice versa, the encoder approximates P_Z from P_X by the distribution $P_{\tilde{Z}}$ of the random variable $\tilde{Z} \coloneqq E(X) + \xi$ with $\xi \sim \mathcal{N}(0, \sigma_z^2 I_d)$. Now, decoder and encoder are trained such that we have $P_X \approx P_{\tilde{X}}$ and $P_Z \approx P_{\tilde{Z}}$. To this end, we aim to maximize the log-likelihood function $\ell(\theta) = \sum_{i=1}^N \log(p_{\tilde{X}}(x_i))$, where θ denotes the parameters D and E depend upon.

The log-density $\log(p_{\tilde{X}}(x))$ induced by the model is called the evidence. However, for VAEs the computation of the evidence is intractable. Therefore, Kingma and Welling (2014) suggest to approximate it by the *evidence lower bound* given by

$$\mathrm{ELBO}(x|\theta) \coloneqq \mathbb{E}_{\xi\sim\mathcal{N}(0,I_d)}[\log(p_Z(E(x)+\sigma_z\xi)) - \tfrac{1}{2\sigma_x^2}\|D(E(x)+\sigma_z\xi) - x\|^2].$$

For the sake of completeness, we include its derivation in Appendix A. Finally, a VAE is trained by minimizing the loss function which sums up the negative ELBO values of all data points, i.e.,

$$\mathcal{L}_{\mathrm{VAE}}(\theta) = -\sum_{i=1}^{N} \mathrm{ELBO}(x_i|\theta).$$

Learned Latent Space. It is a known issue of VAEs that the inferred probability distribution is often more blurry than the ground truth distribution of the data. A detailed discussion of

this issue can be found in Section 2.8.2 of the survey paper by Kingma and Welling (2019). As a remedy, the authors suggest to choose a more flexible model. One possibility is to combine VAEs with normalizing flows, as proposed by Rezende and Mohamed (2015) or Dai and Wipf (2019). Following these approaches, we increase the flexibility of the model by using a latent space learned by a normalizing flow. The idea is based on the observation that transforming probability distributions in low-dimensional spaces is much cheaper than in high-dimensional spaces. Consequently, modelling the low-dimensional latent space can be more effective than learning probability transformations in the high-dimensional data space. Here, we employ the specific loss function proposed by Hagemann et al. (2022, 2023) for training the arising model. More precisely, we choose the latent distribution

$$P_Z = \mathcal{T}_\# P_\Xi,$$

where $\mathcal{T}\colon \mathbb{R}^d \to \mathbb{R}^d$ is an invertible neural network, called normalizing flow. In this way, P_Z is the push-forward of a fixed (known) distribution P_Ξ. Then, the density p_Z is given by

$$p_Z(z) = p_\Xi(\mathcal{T}^{-1}(z))|\det(\nabla \mathcal{T}^{-1}(z))|.$$

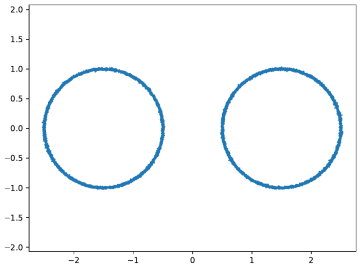
The parameters of $\mathcal{T}$ are considered as trainable parameters. Then, the ELBO reads as

$$\begin{aligned}\mathrm{ELBO}(x|\theta) \coloneqq \mathbb{E}_{\xi\sim\mathcal{N}(0,I_d)}[&\log(p_\Xi(\mathcal{T}^{-1}(E(x)+\sigma_z\xi))) \\ &+ \log(|\det(\nabla \mathcal{T}^{-1}(E(x)+\sigma_z\xi))|) - \tfrac{1}{2\sigma_x^2}\|D(E(x)+\sigma_z\xi)-x\|^2],\end{aligned} \tag{1}$$

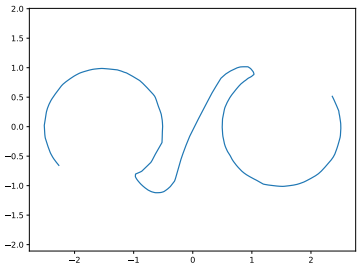
where θ are the parameters of the decoder, the encoder and of the normalizing flow $\mathcal{T}$.

In the literature, there exist several invertible neural network architectures based on coupling blocks (Dinh et al., 2016; Kingma and Dhariwal, 2018), residual networks (Behrmann et al., 2019; Chen et al., 2019; Hertrich, 2023), ODE representations (Chen et al., 2018; Grathwohl et al., 2018; Onken et al., 2021) and autoregressive flows (Huang et al., 2018). In our numerics, we use the coupling-based architecture proposed by Ardizzone et al. (2019).

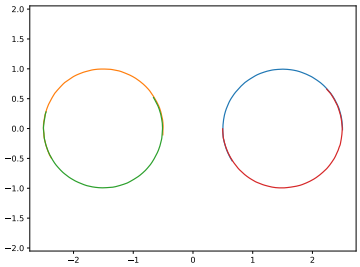
Manifold Learning with VAEs. In order to obtain a lower-dimensional representation of the data points, some papers propose to approximate the data-manifold by $\mathcal{M} \coloneqq \{D(z) : z \in \mathbb{R}^d\}$ (see, e.g., Alberti et al., to appear; Chen et al., 2020; Duff et al., 2021; González et al., 2022). However, this is only possible if the data-manifold admits a global parameterization, i.e., it can be approximated by one generating function. This assumption is often violated in practice. As a toy example, consider the one-dimensional manifold embedded in $\mathbb{R}^2$ that consists of two circles, see Figure 1a. This manifold is disconnected and contains "holes". Consequently, the topologies of the manifold and of the latent space $\mathbb{R}$ do not coincide, so that the manifold cannot be approximated by a VAE. Indeed, this can be verified numerically. When we learn a VAE for approximating samples from this manifold, we observe that the two (generated) circles are not closed and that both components are connected, see Figure 1b. As a remedy, in the next section, we propose the use of multiple generators to resolve this problem, see Figure 1c. For this purpose, we need the notion of charts and atlases.



(a) Noisy samples from the manifold.



(b) Approximation with one chart.



(c) Approximation with four charts.

Figure 1: Example of a one-dimensional manifold that admits no global parameterization.

2.2 Embedded Manifolds

A subset $\mathcal{M} \subseteq \mathbb{R}^n$ is called a d-dimensional embedded differentiable manifold if there exists a family $(U_k, \varphi_k)_{k\in I}$ of relatively open sets $U_k \subseteq \mathcal{M}$ with $\bigcup_{k\in I} U_k = \mathcal{M}$ and mappings $\varphi_k \colon U_k \to \mathbb{R}^d$ such that for every $k, l \in I$

- φ_k is a homeomorphism between U_k and $\varphi_k(U_k)$;
- the inverse $\varphi_k^{-1} \colon \varphi_k(U_k) \to U_k$ is continuously differentiable;
- the transition map $\varphi_k \circ \varphi_l^{-1} \colon \varphi_l(U_k \cap U_l) \to \varphi_k(U_k \cap U_l)$ is continuously differentiable;
- and the Jacobian $\nabla\varphi_k^{-1}(x)$ of φ_k^{-1} at x has full column-rank for any $x \in \varphi_k(U_k)$.

We call the mappings φ_k charts and the family $(U_k, \varphi_k)_{k\in I}$ an atlas. With an abuse of notation, we sometimes also call the set U_k or the pair (U_k, φ_k) a chart. Every compact manifold admits an atlas with finitely many charts $(U_k, \varphi_k)_{k=1}^K$, by definition of compactness.

3. Chart Learning by Mixtures of VAEs

In order to approximate (embedded) manifolds with arbitrary (unknown) topology, we propose to learn several local parameterizations of the manifold instead of a global one. To this end, we propose to use mixture models of VAEs.

An Atlas as Mixture of VAEs. In this paper, we propose to learn the atlas of an embedded manifold $\mathcal{M}$ by representing it as a mixture model of variational autoencoders with decoders $D_k \colon \mathbb{R}^d \to \mathbb{R}^n$, encoders $E_k \colon \mathbb{R}^n \to \mathbb{R}^d$ and normalizing flows $\mathcal{T}_k$ in the latent space, for $k = 1, \dots, K$. Then, the inverse of each chart φ_k will be represented by $\varphi_k^{-1} = \mathcal{D}_k \coloneqq D_k \circ \mathcal{T}_k$. Similarly, the chart φ_k itself is represented by the mapping $\mathcal{E}_k \coloneqq \mathcal{T}_k^{-1} \circ E_k$ restricted to the manifold. Throughout this paper, we denote the parameters of $(D_k, E_k, \mathcal{T}_k)$ by θ_k. Now, let $\tilde{X}_k$, $k = 1, \dots, K$, be the random variable generated by the decoder D_k as in the previous section. Then, we approximate the distribution P_X of the noisy samples from the manifold by the random variable $\tilde{X} \coloneqq \tilde{X}_J$, where J is a discrete random variable on $\{1, \dots, K\}$ with $P(J = k) = \alpha_k$ with mixing weights $\alpha_k > 0$ fulfilling $\sum_{k=1}^K \alpha_k = 1$.

3.1 Training of Mixtures of VAEs

Loss function. Let $x_1, \dots, x_N$ be the noisy training samples. We initialize the weights α by $\alpha_k = \frac{1}{K}$ for all k. They will be estimated later in the training algorithm (see Algorithm 1). In order to train mixtures of VAEs, we again minimize an approximation of an upper bound to the negative log likelihood function $-\sum_{i=1}^N \log(p_{\tilde{X}}(x_i))$. To this end, we employ the law of total probability and Jensen's inequality to obtain

$$\begin{aligned}\log(p_{\tilde{X}}(x_i)) &= \log\Big(\sum_{k=1}^K \beta_{ik} p_{\tilde{X}}(x_i)\Big) \geq \log\Big(\sum_{k=1}^K \beta_{ik}\alpha_k p_{\tilde{X}_k}(x_i)\Big) \\ &\geq \sum_{k=1}^K \beta_{ik}\big(\log(p_{\tilde{X}_k}(x_i)) + \log(\alpha_k)\big) = \sum_{k=1}^K \beta_{ik}\log(p_{\tilde{X}_k}(x_i)) - \log(K)\end{aligned}$$

where $\beta_{ik} \coloneqq P(J = k|\tilde{X} = x_i)$ is the probability that the sample x_i was generated by the k-th random variable $\tilde{X}_k$ and we used that $\alpha_k = \frac{1}{K}$. Using the definition of conditional probabilities, we observe that

$$\beta_{ik} = P(J = k|\tilde{X} = x_i) = \frac{P(J = k)p_{\tilde{X}_k}(x_i)}{p_{\tilde{X}}(x_i)} = \frac{\alpha_k p_{\tilde{X}_k}(x_i)}{\sum_{j=1}^K \alpha_j p_{\tilde{X}_j}(x_i)}. \tag{2}$$

As the computation of $p_{\tilde{X}_k}$ is intractable, we replace it by the ELBO (1), i.e., we approximate β_{ik} by

$$\tilde{\beta}_{ik} = \frac{\alpha_k \exp(\mathrm{ELBO}(x_i|\theta_k))}{\sum_{j=1}^K \alpha_j \exp(\mathrm{ELBO}(x_i|\theta_j))}. \tag{3}$$

For the architecture used in our numerical examples in Section 5, we can bound the error introduced by this approximation as in Corollary 14 of Appendix B. More precisely, we show that there exists some L such that $\frac{1}{L}\beta_{ik} \leq \tilde{\beta}_{ik} \leq L\beta_{ik}$. Then, we arrive at the approximation

$$\log(p_{\tilde{X}}(x_i)) \approx \ell(x_i|\Theta) = \sum_{k=1}^K \frac{\alpha_k \exp(\mathrm{ELBO}(x_i|\theta_k))}{\sum_{j=1}^K \alpha_j \exp(\mathrm{ELBO}(x_i|\theta_j))}\mathrm{ELBO}(x_i|\theta_k) - \log(K).$$

By summing up over all i, we approximate the negative log likelihood function by the loss function

$$\mathcal{L}(\Theta) = -\sum_{i=1}^N \ell(x_i|\Theta),$$

in order to train the parameters $\Theta = (\theta_k)_{k=1}^K$ of the mixture of VAEs. Finally, this loss function is then optimized with a stochastic gradient based optimizer like Adam (Kingma and Ba, 2015).

Remark 1 (Lipschitz regularization) *In order to represent the local structure of the manifold and to stabilize the training, we would like to avoid that two points that are close in the latent distribution have too large a distance in the data space. This corresponds to regularizing the Lipschitz constant of the decoders D_k and of the normalizing flows $\mathcal{T}_k$. More precisely, for some small $\sigma > 0$, we add the regularization term*

$$\mathcal{R}(\Theta) \coloneqq \frac{1}{\sigma^2}\sum_{k=1}^K \mathbb{E}_{z\sim P_\Xi, \eta\sim\mathcal{N}(0,\sigma^2)}\Big[D_k(\mathcal{T}_k(z)) - D_k(\mathcal{T}_k(z+\eta))\Big]$$

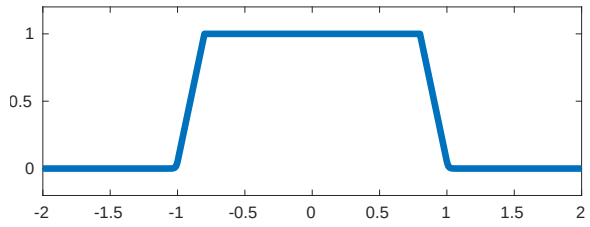


Figure 2: Plot of the unnormalized latent density q.

for the first few epochs of the training. Once the charts roughly capture the local structures of the manifold, we avoid the Lipschitz regularization in order to have the interpretation of the loss function as an approximation of the negative log likelihood of the training points.

Latent Distribution. In order to identify the sets U_k defining the domain of the k-th learned chart, we choose a latent distribution that is mostly concentrated in the rectangle $(-1,1)^d$. Then, we can define the domain U_k of the k-th learned chart as the set $U_k \coloneqq \mathcal{D}_k((-1,1)^d)$. Since the charts are supposed to overlap, the density should become small close to the boundary. To this end, we choose the distribution P_Ξ by using the density $p_\Xi(z) \coloneqq \prod_{i=1}^d q(z_i)$, where the density q is up to a multiplicative constant given by

$$q(z) \propto \begin{cases} 1, & |z| < 0.8, \\ 4.8 - 4.75|z|, & |z| \in [0.8, 1], \\ 0.05\exp(-100(|z|-1)), & |z| > 1, \end{cases}$$

see Figure 2 for a plot.

Due to approximation errors and noise, we will have to deal with points $x \in \mathbb{R}^n$ that are not exactly located in one of the sets U_k. In this case, we cannot be certain to which charts the point x actually belongs. Therefore, we interpret the conditional probability (2) as the probability that x_i belongs to the k-th chart. Since we cannot compute the β_{ik} explicitly, we use the approximations $\tilde{\beta}_{ik}$ from (3) instead.

Overlapping Charts. Since the sets U_k of an atlas $(U_k, \varphi_k)_{k\in I}$ are an open covering of the manifold, they have to overlap near their boundaries. To this end, we propose the following post-processing heuristic.

By the definition of the loss function $\mathcal{L}$, we have that the k-th generator $\mathcal{D}_k$ is trained such that U_k contains all points x_i from the training set where $\tilde{\beta}_{ik}$ is non-zero. The following procedure modifies the $\tilde{\beta}_{ik}$ in such a way that the samples x that are close to the boundary of the k-th chart will also be assigned to a second chart.

Since the measure P_Ξ is mostly concentrated in $(-1,1)^d$, the region close to the boundary of the k-th chart can be identified by $D_k(\mathcal{T}_k(z))$ for all z close to the boundary of $(-1,1)^d$. For $c > 1$, we define the modified ELBO function

$$\begin{aligned}\mathrm{ELBO}_c(x|\theta_k) \coloneqq \mathbb{E}_{\xi\sim\mathcal{N}(0,I_d)}[&\log(p_\Xi(c\mathcal{T}_k^{-1}(E_k(x)+\sigma_z\xi))) \\ &+ \log(|\det(\nabla\mathcal{T}_k^{-1}(E_k(x)+\sigma_z\xi))|) - \tfrac{1}{2\sigma_x^2}\|D_k(E_k(x)+\sigma_z\xi) - x\|^2]\end{aligned}$$

which differs from (1) by the additional scaling factor c within the first summand. By construction and by the definiton of p_Ξ, it holds that $\mathrm{ELBO}_c(x,\theta_k) \approx \mathrm{ELBO}(x|\theta_k)$ whenever $c\|\mathcal{T}_k^{-1}(E_k(x))\|_\infty < 0.8$ and $0 < \sigma_z \ll 0.1$ is small. Otherwise, we have $\mathrm{ELBO}_c(x|\theta_k) < \mathrm{ELBO}(x|\theta_k)$ In particular, $\mathrm{ELBO}_c(x|\theta_k)$ is close to $\mathrm{ELBO}(x|\theta_k)$ if x belongs to the interior of the k-th chart and is significantly smaller if it is close to the boundary of the k-th chart.

As a variation of $\tilde{\beta}_{ik}$, now we define

$$\hat{\gamma}_{il}^{(l)} = \frac{\alpha_l \exp(\mathrm{ELBO}_c(x_i|\theta_l))}{\alpha_l \exp(\mathrm{ELBO}_c(x_i|\theta_l)) + \sum_{j\in\{1,\dots,K\}\setminus\{l\}} \alpha_j \exp(\mathrm{ELBO}(x_i|\theta_j))}$$

and

$$\hat{\gamma}_{ik}^{(l)} = \frac{\alpha_k \exp(\mathrm{ELBO}(x_i|\theta_k))}{\alpha_l \exp(\mathrm{ELBO}_c(x_i|\theta_l)) + \sum_{j\in\{1,\dots,K\}\setminus\{l\}} \alpha_j \exp(\mathrm{ELBO}(x_i|\theta_j))}$$

for $k,l \in \{1,\dots,K\}$. Similarly as the $\tilde{\beta}_{ik}$, $\hat{\gamma}_{ik}^{(l)}$ can be viewed as a classification parameter, which assigns each x_i either to a chart $k \neq l$ or to the interior part of the l-th chart. Consequently, points located near the boundary of the l-th chart will also be assigned to another chart. Finally, we combine and normalize the $\hat{\gamma}_{ik}^{(l)}$ by

$$\gamma_{ik} = \frac{\hat{\gamma}_{ik}}{\sum_{k=1}^K \hat{\gamma}_{ik}}, \quad \text{where} \quad \hat{\gamma}_{ik} = \max_{l=1,\dots,K} \hat{\gamma}_{ik}^{(l)}. \tag{4}$$

Once, the γ_{ik} are computed, we update the mixing weights α by $\alpha_k = \sum_{i=1}^N \gamma_{ik}$ and minimize the loss function

$$\mathcal{L}_{\mathrm{overlap}}(\Theta) = -\sum_{i=1}^N \sum_{k=1}^K \gamma_{ik} \mathrm{ELBO}(x_i|\theta_k) \tag{5}$$

using a certain number of epochs of the Adam optimizer (Kingma and Ba, 2015).

The whole training procedure for a mixture of VAEs representing the charts of an embedded manifold is summarized in Algorithm 1. The hyperparameters M_1, M_2 and M_3 are chosen large enough such that we have approximately approached a local minimum of the corresponding objective function. In our numerical examples, we choose $M_1 = 50$, $M_2 = 150$ and $M_3 = 50$.

Remark 2 (Number of charts K) *The choice of the number of charts K is a trade-off between the computational efficiency and the flexibility of the model. While each manifold has a minimal number of required charts, it could be easily represented by more charts. Therefore, from a topological viewpoint, there is no upper limit on K. However, since each chart comes with its own pair of decoder and encoder, the number of parameters within the mixture of VAEs, and consequently the training effort, grow with K.*

Algorithm 1 Training procedure for mixtures of VAEs.

1. Run the Adam optimizer on $\mathcal{L}(\Theta) + \lambda\mathcal{R}(\Theta)$ for M_1 epochs.
2. Run the Adam optimizer on $\mathcal{L}(\Theta)$ for M_2 epochs.
3. Compute the values γ_{ik}, $i = 1, \dots, N$, $k = 1, \dots, K$ from (4).
4. Compute the mixing weights $\alpha_k = \sum_{i=1}^{N} \gamma_{ik}$.
5. Run the Adam optimizer on $\mathcal{L}_{\text{overlap}}(\Theta)$ from (5) for M_3 epochs.

3.2 Architectures

In this subsection, we focus on the architecture of the VAEs used in the mixture model representing the manifold $\mathcal{M}$. Since $\mathcal{D}_k = D_k \circ \mathcal{T}_k$ represent the inverse of our charts φ_k, the decoders have to be injective. Moreover, since $\mathcal{E}_k = \mathcal{T}_k^{-1} \circ E_k$ represents the chart itself, the condition $\mathcal{E}_k \circ \mathcal{D}_k = \mathrm{Id}$ must be verified. Therefore, we choose the encoder E_k as a left-inverse of the corresponding decoder D_k. More precisely, we use the decoders of the form

$$D_k = T_L \circ A_L \circ \cdots \circ T_1 \circ A_1,$$

where the $T_l \colon \mathbb{R}^{d_l} \to \mathbb{R}^{d_l}$ are invertible neural networks and $A_l \colon \mathbb{R}^{d_{l-1}} \to \mathbb{R}^{d_l}$ are fixed injective linear operators for $l = 1, \dots, L$, $d = d_0 < d_1 < \cdots < d_L = n$. As it is a concatenation of injective mappings, we obtain that D_k is injective. Finally, the corresponding encoder is given by

$$E_k = A_1^\dagger \circ T_1^{-1} \circ \cdots \circ A_L^\dagger \circ T_L^{-1}, \quad A^\dagger = (A^{\mathrm{T}} A)^{-1} A^{\mathrm{T}}. \tag{6}$$

Then, it holds by construction that $\mathcal{E}_k \circ \mathcal{D}_k = \mathrm{Id}$.

In this paper, we build the invertible neural networks T_l and the normalizing flows $\mathcal{T}_k$ out of coupling blocks as proposed by Ardizzone et al. (2019) based on the real NVP architecture (Dinh et al., 2016). To this end, we split the input $z \in \mathbb{R}^{d_l}$ into two parts $z = (z_1, z_2) \in \mathbb{R}^{d_l^1} \times \mathbb{R}^{d_l^2}$ with $d_l = d_l^1 + d_l^2$ and define $T_l(z) = (x_1, x_2)$ with

$$x_1 = z_1 \, \mathrm{e}^{s_2(z_2)} + t_2(z_2) \quad \text{and} \quad x_2 = z_2 \, \mathrm{e}^{s_1(x_1)} + t_1(x_1),$$

where $s_1, t_1 \colon \mathbb{R}^{d_l^1} \to \mathbb{R}^{d_l^2}$ and $s_2, t_2 \colon \mathbb{R}^{d_l^2} \to \mathbb{R}^{d_l^1}$ are arbitrary subnetworks (depending on l). Then, the inverse $T_l^{-1}(x_1, x_2)$ can analytically be derived as $z = (z_1, z_2)$ with

$$z_2 = \big(x_2 - t_1(x_1)\big) \, \mathrm{e}^{-s_1(x_1)} \quad \text{and} \quad z_1 = \big(x_1 - t_2(z_2)\big) \, \mathrm{e}^{-s_2(z_2)}.$$

Remark 3 (Projection onto learned charts) *Consider a decoder D_k and an encoder E_k as defined above. By construction, the mapping $\pi_k = D_k \circ E_k$ is a (nonlinear) projection onto* $\mathrm{range}(D_k) = \mathrm{range}(\pi_k)$*, in the sense that $\pi_k \circ \pi_k = \pi_k$ and that $\pi_{k|\mathrm{range}(D_k)}$ is the identity on* $\mathrm{range}(D_k)$*. Consequently, the mapping π_k is a projection on the range of D_k which represents the k-th chart of $\mathcal{M}$. In particular, there is an open neighborhood $V := \pi_k^{-1}(U_k) \subseteq \mathbb{R}^n$ such that $\pi_{k|V}$ is a projection onto U_k.*

4. Optimization on Learned Manifolds

As motivated in the introduction, we are interested in optimization problems of the form

$$\min_{x \in \mathbb{R}^n} F(x) \quad \text{subject to} \quad x \in \mathcal{M}, \tag{7}$$

where $F\colon \mathbb{R}^n \to \mathbb{R}$ is a differentiable function and $\mathcal{M}$ is available only through some data points. In the previous section, we proposed a way to represent the manifold $\mathcal{M}$ by a mixture model $(D_k, E_k, \mathcal{T}_k)$ of VAEs. This section outlines a gradient descent algorithm for the solution of (7) once the manifold is learned.

As outlined in the previous section, the inverse charts φ_k^{-1} of the manifold $\mathcal{M}$ are modeled by $\mathcal{D}_k := D_k \circ \mathcal{T}_k$. The chart φ_k itself is given by the mapping $\mathcal{E}_k := \mathcal{T}_k^{-1} \circ E_k$ restricted to the manifold. For the special case of a VAE with a single generator $\mathcal{D}$, Alberti et al. (to appear); Chen et al. (2020); Duff et al. (2021) propose to solve (7) in the latent space. More precisely, starting with a latent initialization $z_0 \in \mathbb{R}^d$ they propose to solve

$$\min_{z \in \mathbb{R}^d} F(\mathcal{D}(z))$$

using a gradient descent scheme. However, when using multiple charts, such a gradient descent scheme heavily depends on the current chart. Indeed, the following example shows that the gradient direction can change significantly, if we use a different chart.

Example 1 *Consider the two-dimensional manifold $\mathbb{R}^2$ and the two learned charts given by the generators*

$$\mathcal{D}_1(z_1, z_2) = (10z_1, z_2), \quad \textit{and} \quad \mathcal{D}_2(z_1, z_2) = (z_1, 10z_2).$$

Moreover let $F\colon \mathbb{R}^2 \to \mathbb{R}$ be given by $(x, y) \mapsto x + y$. Now, the point $x^{(0)} = (0, 0)$ corresponds for both charts to $z^{(0)} = (0, 0)$. A gradient descent step with respect to $F \circ \mathcal{D}_k$, $k = 1, 2$, using step size τ yields the latent values

$$\begin{aligned}(z_1^{(1)}, z_2^{(1)}) &= z^{(0)} - \tau \nabla (F \circ \mathcal{D}_1)(z^{(0)}) = -(10\tau, \tau),\\ (\tilde{z}_1^{(1)}, \tilde{z}_2^{(1)}) &= z^{(0)} - \tau \nabla (F \circ \mathcal{D}_2)(z^{(0)}) = -(\tau, 10\tau).\end{aligned}$$

Thus, one gradient steps with respect to $F \circ \mathcal{D}_k$ yields the values

$$x^{(1)} = \mathcal{D}_1(z^{(1)}) = -(100\tau, \tau), \quad \tilde{x}^{(1)} = \mathcal{D}_2(\tilde{z}^{(1)}) = -(\tau, 100\tau).$$

Consequently, the gradient descent steps with respect to two different charts can point into completely different directions, independently of the step size τ.

Therefore, we aim to use a gradient formulation which is independent of the parameterization of the manifold. Here, we use the concept of the Riemannian gradient with respect to the Riemannian metric, which is inherited from the Euclidean space in which the manifold $\mathcal{M}$ is embedded. To this end, we first revisit some basic facts about Riemannian gradients on embedded manifolds which can be found, e.g., in the book of Absil et al. (2009). Afterwards, we consider suitable retractions in order to perform a descent step in the direction of the negative Riemannian gradient. Finally, we use these notions in order to derive a gradient descent procedure on a manifold given by mixtures of VAEs.

4.1 Background on Riemannian Optimization

Riemannian Gradients on Embedded Manifolds. Let $x \in \mathcal{M} \subseteq \mathbb{R}^n$ be a point on the manifold, let $\varphi\colon U \to \mathbb{R}^d$ be a chart with $x \in U$ and $\varphi^{-1}\colon \varphi(U) \to U$ be its inverse. Then, the tangent space is given by the set of all directions $\dot\gamma(0)$ of differentiable curves $\gamma\colon (-\epsilon,\epsilon) \to \mathcal{M}$ with $\gamma(0) = x$. More precisely, it is given by the linear subspace of $\mathbb{R}^n$ defined as

$$T_x\mathcal{M} = \{Jy : y \in \mathbb{R}^d\}, \quad \text{where } J \coloneqq \nabla\varphi^{-1}(\varphi(x)) \in \mathbb{R}^{n\times d}. \tag{8}$$

The tangent space inherits the Riemannian metric from $\mathbb{R}^n$, i.e., we equip the tangent space with the inner product

$$\langle u, v\rangle_x = u^{\mathrm{T}} v, \quad u, v \in T_x\mathcal{M}.$$

A function $f\colon \mathcal{M} \to \mathbb{R}^m$ is called differentiable if for any differentiable curve $\gamma\colon (-\epsilon,\epsilon) \to \mathcal{M}$ we have that $f \circ \gamma\colon (-\epsilon,\epsilon) \to \mathbb{R}^m$ is differentiable. In this case the differential of f is defined by

$$Df(x)\colon T_x\mathcal{M} \to \mathbb{R}^m, \qquad Df(x)[h] = \frac{\mathrm{d}}{\mathrm{d}t} f(\gamma_h(t))\Big|_{t=0},$$

where $\gamma_h\colon (-\epsilon,\epsilon) \to \mathcal{M}$ is a curve with $\gamma_h(0) = x$ and $\dot\gamma_h(0) = h$. Finally, the Riemannian gradient of a differentiable function $f\colon \mathcal{M} \to \mathbb{R}$ is given by the unique element $\nabla_{\mathcal{M}} f \in T_x\mathcal{M}$ which fulfills

$$Df(x)[h] = \langle \nabla_{\mathcal{M}} f, h\rangle_x \quad \text{for all} \quad h \in T_x\mathcal{M}.$$

Remark 4 *In the case that f can be extended to a differentiable function on a neighborhood of $\mathcal{M}$, these formulas can be simplified. More precisely, we have that the differential is given by $Df(x)[h] = h^{\mathrm{T}}\nabla f(x)$, where ∇f is the Euclidean gradient of f. In other words, $Df(x)$ is the Fréchet derivative of f at x restricted to $T_x\mathcal{M}$. Moreover, the Riemannian gradient coincides with the orthogonal projection of ∇f on the tangent space, i.e.,*

$$\nabla_{\mathcal{M}} f(x) = P_{T_x\mathcal{M}}(\nabla f(x)), \quad P_{T_x\mathcal{M}}(y) = \operatorname*{arg\,min}_{z\in T_x\mathcal{M}} \|y - z\|^2.$$

Here the orthogonal projection can be rewritten as $P_{T_x\mathcal{M}} = J(J^{\mathrm{T}}J)^{-1}J^{\mathrm{T}}$, $J = \nabla\varphi^{-1}(\varphi(x))$ such that the Riemannian gradient is given by $\nabla_{\mathcal{M}} f(x) = J(J^{\mathrm{T}}J)^{-1}J^{\mathrm{T}}\nabla f(x)$.

Retractions. Once the Riemannian gradient is computed, we aim to perform a descent step in the direction of $-\nabla_{\mathcal{M}} f(x)$ on $\mathcal{M}$. To this end, we need the concept of retraction. Roughly speaking, a retraction in x maps a tangent vector ξ to the point that is reached by moving from x in the direction ξ. Formally, it is defined as follows.

Definition 5 *A differentiable mapping $R_x\colon V_x \to \mathcal{M}$ for some neighborhood $V_x \subseteq T_x\mathcal{M}$ of 0 is called a retraction in x, if $R_x(0) = x$ and*

$$DR_x(0)[h] = h \quad \textit{for all} \quad h \in T_0(V_x) = T_x\mathcal{M},$$

where we identified $T_0(V_x)$ with $T_x\mathcal{M}$. Moreover, a differentiable mapping $R = (R_x)_{x\in\mathcal{M}}\colon V \to \mathcal{M}$ on a subset of the tangent bundle $V = (V_x)_{x\in\mathcal{M}} \subseteq T\mathcal{M} = (T_x\mathcal{M})_{x\in\mathcal{M}}$ is a retraction on $\mathcal{M}$, if for all $x \in \mathcal{M}$ we have that R_x is a retraction in x.

Now, let $R\colon V \to \mathcal{M}$ be a retraction on $\mathcal{M}$. Then, the Riemannian gradient descent scheme starting at $x_0 \in \mathcal{M}$ with step size $\tau > 0$ is defined by

$$x_{t+1} = R_{x_t}(-\tau \nabla_{\mathcal{M}} f(x_t)).$$

4.2 Retractions for Learned Charts

In order to apply this gradient scheme for a learned manifold given by the learned mappings $(\mathcal{D}_k, \mathcal{E}_k)_{k=1}^K$, we consider two types of retractions. We introduce them and show that they are indeed retractions in the following lemmas. The first one generalizes the idea from Lemma 4 and Proposition 5 in Absil and Malick (2012) of moving along the tangent vector in $\mathbb{R}^n$ and reprojecting onto the manifold. However, the results by Absil and Malick (2012) are based on the orthogonal projection, which is hard or even impossible to compute. Thus, we replace it by some more general projection π. In our applications, π will be chosen as in Remark 3, i.e., we set $\pi(x) = \mathcal{D}_k(\mathcal{E}_k(x))$.

Lemma 6 *Let $x \in \mathcal{M}$, $U_x \subseteq \mathbb{R}^n$ be a neighborhood of x in $\mathbb{R}^n$, $\pi\colon U_x \to \mathcal{M} \cap U_x$ be a differentiable map such that $\pi \circ \pi = \pi$. Set $V_x = \{h \in T_x\mathcal{M} \subseteq \mathbb{R}^n : x + h \in U_x\}$. Then*

$$R_x(h) = \pi(x + h), \qquad h \in V_x,$$

defines a retraction in x.

Proof The property $R_x(0) = x$ is directly clear from the definition of R_x. Now let $h \in T_x\mathcal{M} \subseteq \mathbb{R}^n$ and $\gamma_h\colon (-\epsilon, \epsilon) \to \mathcal{M}$ be a differentiable curve with $\gamma_h(0) = x$ and $\dot{\gamma}_h(0) = h$. As $\pi_{|U}$ is the identity on $\mathcal{M}$, we have by the chain rule that

$$h = \dot{\gamma}_h(t) = \frac{\mathrm{d}}{\mathrm{d}t}\pi(\gamma_h(t))\Big|_{t=0} = \nabla\pi(x)\dot{\gamma}_h(0) = \nabla\pi(x)h,$$

where $\nabla\pi(x)$ is the Euclidean Jacobian matrix of π at x. Similarly,

$$DR_x(0)[h] = \frac{\mathrm{d}}{\mathrm{d}t}R_x(th)\Big|_{t=0} = \frac{\mathrm{d}}{\mathrm{d}t}\pi(x + th)\Big|_{t=0} = \nabla\pi(x)h = h.$$

■

The second retraction uses the idea of changing to local coordinates, moving into the gradient direction by using the local coordinates and then going back to the manifold representation. Note that similar constructions are considered in Section 4.1.3 in the book of Absil et al. (2009). However, as we did not find an explicit proof for the lemma, we give it below for the sake of completeness.

Lemma 7 *Let $x \in \mathcal{M}$ and denote by $\varphi\colon U \to \mathbb{R}^d$ a chart with $x \in U \subseteq \mathcal{M}$. Then,*

$$R_x(h) = \varphi^{-1}(\varphi(x) + (J^{\mathrm{T}}J)^{-1}J^{\mathrm{T}}h), \quad J = \nabla\varphi^{-1}(\varphi(x))$$

defines a retraction in x.

Algorithm 2 One gradient descent step on a learned manifold.

Inputs: Function $F\colon \mathcal{M} \to \mathbb{R}$, point x_n, step size $\tau_n > 0$.
for $k = 1, \dots, K$ **do**
- Approximate the probability that x_n belongs to chart k by computing the β_k from (10).
- Project to the k-th chart by $\tilde{x}_{n,k} = \mathcal{D}_k(\mathcal{E}_k(x_n))$.
- Compute the Riemannian gradient $g_{n,k} = \nabla_{\mathcal{M}} F(\tilde{x}_{n,k})$, e.g., by Remark 4.
- Perform a gradient descent with the retraction $R_{k,\tilde{x}_{n,k}}$, i.e., define $x_{n+1,k} = R_{k,\tilde{x}_{n,k}}(-\tau_n g_{n,k})$.

end for
- Average results by computing $x_{n+1} = \sum_{k=1}^{K} \beta_k x_{n+1,k}$.

Proof The property $R_x(0) = x$ is directly clear from the definition of R_x. Now let $h \in T_0(T_x\mathcal{M}) = T_x\mathcal{M} \subseteq \mathbb{R}^n$. By (8), we have that there exists some $y \in \mathbb{R}^d$ such that $h = Jy$. Then, we have by the chain rule that

$$DR_x(0)[h] = \frac{\mathrm{d}}{\mathrm{d}t} R_x(th)\Big|_{t=0} = (\nabla\varphi^{-1}(\varphi(x)))(J^{\mathrm{T}}J)^{-1}J^{\mathrm{T}}h = J(J^{\mathrm{T}}J)^{-1}J^{\mathrm{T}}Jy = Jy = h.$$

■

4.3 Gradient Descent on Learned Manifolds

By Lemma 6 and 7, we obtain that the mappings

$$R_{k,x}(h) = \mathcal{D}_k(\mathcal{E}_k(x+h)) \quad \text{and} \quad \tilde{R}_{k,x}(h) = \mathcal{D}_k(\mathcal{E}_k(x) + (J^{\mathrm{T}}J)^{-1}J^{\mathrm{T}}h) \tag{9}$$

with $J = \nabla\mathcal{D}_k(\mathcal{E}_k(x))$ are retractions in all $x \in U_k$. If we define R such that R_x is given by R_k for some k such that $x \in U_k$, then the differentiability of $R = (R_x)_{x\in\mathcal{M}}$ in x might be violated. Moreover, the charts learned by a mixture of VAEs only overlap approximately and not exactly. Therefore, these retractions cannot be extended to a retraction on the whole manifold $\mathcal{M}$ in general. As a remedy, we propose the following gradient descent step on a learned manifold.

Starting from a point x_n, we first compute for $k = 1, \dots, K$ the probability, that x_n belongs to the k-th chart. By (2), this probability can be approximated by

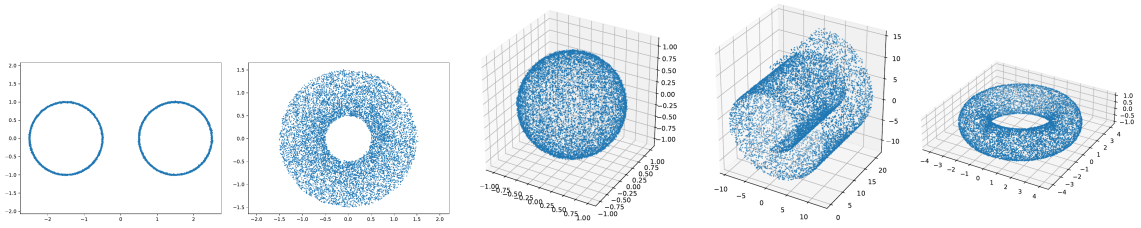
$$\beta_k \coloneqq \frac{\alpha_k \exp(\mathrm{ELBO}(x_n|\theta_k))}{\sum_{j=1}^{K} \alpha_j \exp(\mathrm{ELBO}(x_n|\theta_j))}. \tag{10}$$

Afterwards, we project x_n onto the k-th chart by applying $\tilde{x}_{n,k} = \mathcal{D}_k(\mathcal{E}_k(x_n))$ (see Remark 3) and compute the Riemannian gradient $g_{n,k} = \nabla_{\mathcal{M}} F(\tilde{x}_{n_k})$. Then, we apply the retraction $R_{k,\tilde{x}_{n,k}}$ (or $\tilde{R}_{k,\tilde{x}_{n,k}}$) to perform a gradient descent step $x_{n+1,k} = R_{k,\tilde{x}_{n,k}}(-\tau_n g_{n,k})$. Finally, we average the results by $x_{n+1} = \sum_{k=1}^{K} \beta_k x_{n+1,k}$.

The whole gradient descent step is summarized in Algorithm 2. Finally, we compute the sequence $(x_n)_n$ by applying Algorithm 2 iteratively.

Algorithm 3 Adaptive step size scheme for gradient descent on learned manifolds

Input: Function F, initial point x_0, initial step size τ_0.
for n=0,1,... **do**
  Compute x_{n+1} by Algorithm 2 with step size τ_n.
  while $F(x_{n+1}) > F(x_n)$ **do**
    Update step size by $\tau_n \leftarrow \frac{\tau_n}{2}$.
    Update x_{n+1} by Algorithm 2 with the new step size τ_n.
  end while
  Set step size for the next step $\tau_{n+1} = \frac{3\tau_n}{2}$.
end for



Two circles Ring Sphere Swiss roll Torus

Figure 3: Data sets used for the different manifolds.

For some applications the evaluation of the derivative of F is computationally costly. Therefore, we aim to take as large step sizes τ_n as possible in Algorithm 2. On the other hand, large step sizes can lead to numerical instabilities and divergence. To this end, we use an adaptive step size selection as outlined in Algorithm 3.

Remark 8 (Descent algorithm) *By construction, Algorithm 3 is a descent algorithm. That is, for a sequence $(x_n)_n$ generated by the algorithm it holds that $F(x_{n+1}) \leq F(x_n)$. With the additional assumption that F is bounded from below, we have that $(F(x_n))_n$ is a bounded descending and hence convergent sequence. However, this does neither imply convergence of the iterates $(x_n)_n$ themselves nor optimality of the limit of $(F(x_n))_n$. For more details on the convergence of line-search algorithms on manifolds we refer to Section 4 of the book by Absil et al. (2009).*

5. Numerical Examples

Next, we test the numerical performance of the proposed method. In this section, we start with some one- and two-dimensional manifolds embedded in the two- or three-dimensional Euclidean space. We use the architecture from Section 3.2 with $L = 1$. That is, for all the manifolds, the decoder is given by $\mathcal{T} \circ A$ where $A \colon \mathbb{R}^d \to \mathbb{R}^n$ is given by $x \mapsto (x, 0)$ if $d = n-1$ and by $A = \mathrm{Id}$ if $d = n$ and $\mathcal{T}$ is an invertible neural network with 5 invertible coupling blocks where the subnetworks have two hidden layers and 64 neurons in each layer. The normalizing flow modelling the latent space consists of 3 invertible coupling blocks with the

same architecture. We train the mixture of VAEs for 200 epochs with the Adam optimizer. Afterwards we apply the overlapping procedure for 50 epochs, as in Algorithm 1.

We consider the manifolds "two circles", "ring", "sphere", "swiss roll" and "torus". The (noisy) training data are visualized in Figure 3. The number of charts K is given in the following table.

	Two circles	Ring	Sphere	Swiss roll	Torus
Number of charts	4	2	2	4	6

We visualize the learned charts in Figure 4. Moreover, additional samples generated by the learned mixture of VAEs are shown in Figure 5. We observe that our model covers all considered manifolds and provides a reasonable approximation of different charts. Finally, we test the gradient descent method from Algorithm 2 with some linear and quadratic functions, which often appear as data fidelity terms in inverse problems:

- $F(x) = x_2$ on the manifold "two circles" with initial points $\frac{x_0}{\|x_0\|} \pm (1.5, 0)$ for $x_0 = (\pm 0.2, 1)$;
- $F(x) = \|x - (-1, 0)\|^2$ on the manifold "ring" with initial points $(1, \pm 0.4)$;
- $F(x) = \|x - (0, 0, -2)\|^2$ on the manifold "sphere" with inital points $x_0/\|x_0\|$ for $x_0 \in \{(0.3\cos(\frac{\pi k}{5}), 0.3\sin(\frac{\pi k}{5}), 1) : k = 0, \dots, 9)\}$;
- $F(x) = \|x - (-5, 0, 0)\|^2$ on the manifold "torus", where the inital points are drawn randomly from the training set.

We use the retraction from Lemma 6 with a step size of 0.01. The resulting trajectories are visualized in Figure 6. We observe that all the trajectories behave as expected and approach the closest minimum of the objective function, even if this is not in the same chart of the initial point.

Remark 9 (Dimension of the Manifold) *For all our numerical experiments, we assume that the dimension d of the data manifold is known. This assumption might be violated for practical applications. However, there exist several methods in the literature to estimate the dimension of a manifold from data (see, e.g., Stanczuk et al., 2024; Camastra and Staiano, 2016; Fan et al., 2010; Levina and Bickel, 2004). We are aware that dimension estimation of high dimensional data sets is a hard problem which cannot be considered as completely solved so far. In particular, most of these algorithms make assumptions on the distribution of the given data points. Even though it is an active area of research, it is not the scope of this paper to test, benchmark or develop such algorithms. Similarly, combining them with our mixture of VAEs is left for future research.*

6. Mixture of VAEs for Inverse Problems

In this section we describe how to use mixture of VAEs to solve inverse problems. We consider an inverse problem of the form

$$y = \mathcal{G}(x) + \eta, \tag{11}$$

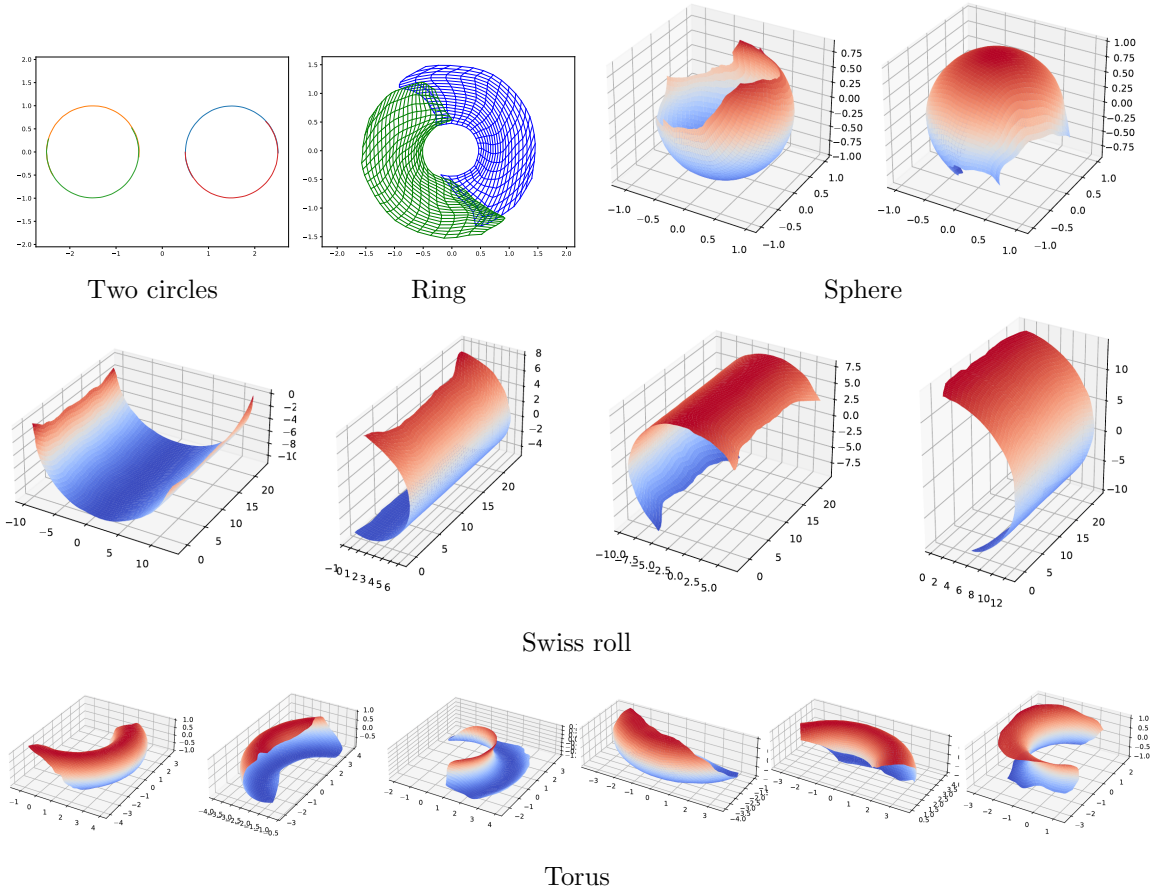


Figure 4: Learned charts for the different manifolds. For the manifolds "two circles" and "ring", each color represents one chart. For the manifolds "sphere", "swiss roll" and "torus" we plot each chart in a separate figure.

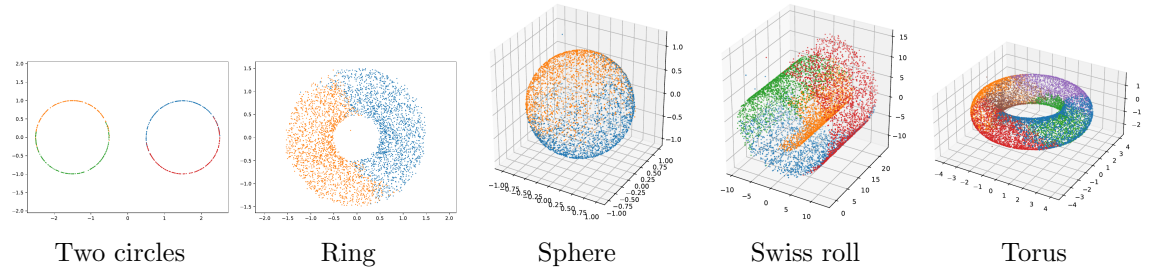
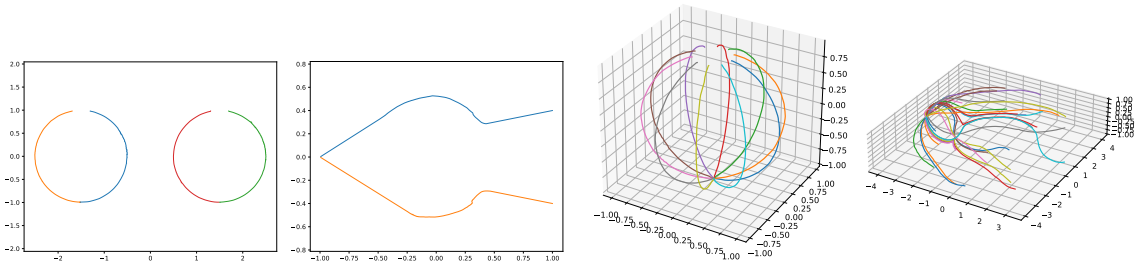


Figure 5: Generated samples by the learned mixture of VAEs. The color of a point indicates from which generator the point was sampled.



Two circles Ring Sphere Torus

Figure 6: Trajectories of the gradient descent on the learned manifolds.

where $\mathcal{G}$ is a possibly nonlinear map between $\mathbb{R}^n$ and $\mathbb{R}^m$, modelling a measurement (forward) operator, $x \in \mathbb{R}^n$ is a quantity to be recovered, $y \in \mathbb{R}^m$ is the noisy data and η represents some noise. In particular, we analyze a linear and a nonlinear inverse problem: a deblurring problem and a parameter identification problem for an elliptic PDE arising in electrical impedance tomography (EIT), respectively.

In many inverse problems, the unknown x can be modeled as an element of a low-dimensional manifold $\mathcal{M}$ in $\mathbb{R}^n$ (Alberti et al., to appear; Asim et al., 2020; Bora et al., 2017; Hyun et al., 2021; Ongie et al., 2020; Seo et al., 2019; Massa et al., 2022; Alberti et al., 2023), and this manifold can be represented by the mixture of VAEs as explained in Section 3. Thus, the solution of (11) can be found by optimizing the function

$$F(x) = \frac{1}{2}\|\mathcal{G}(x) - y\|^2_{\mathbb{R}^m} \quad \text{subject to } x \in \mathcal{M}, \tag{12}$$

by using the iterative scheme proposed in Section 4.

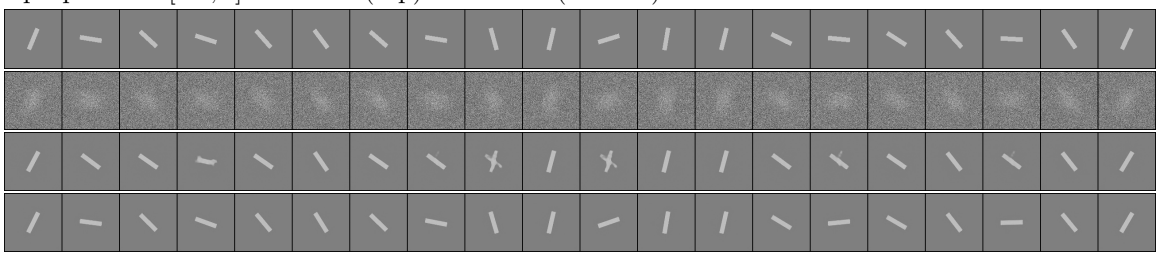
We would like to emphasize that the main goal of our experiments is not to obtain state-of-the-art results. Instead, we want to highlight the advantages of using multiple generators via a mixture of VAEs. All our experiments are designed in such a way that the manifold property of the data is directly clear. The application to real-world data and the combination with other methods in order to achieve competitive results are not within the scope of this paper and are left to future research.

Architecture and Training. Throughout these experiments we consider images of size 128×128 and use the architecture from Section 3.2 with $L = 3$. Starting with the latent dimension d, the mapping $A_1 \colon \mathbb{R}^d \to \mathbb{R}^{32^2}$ fills up the input vector x with zeros up to the size 32^2, i.e., we set $A_1(x) = (x, 0)$. The invertible neural network $T_1 \colon \mathbb{R}^{32^2} \to \mathbb{R}^{32^2}$ consists of 3 invertible blocks, where the subnetworks s_i and t_i, $i = 1, 2$ are dense feed-forward networks with two hidden layers and 64 neurons. Afterwards, we reorder the dimensions to obtain an image of size 32×32. The mappings $A_2 \colon \mathbb{R}^{32\times 32} \to \mathbb{R}^{32\times 32\times 4}$ and $A_3 \colon \mathbb{R}^{64\times 64} \to \mathbb{R}^{64\times 64\times 4}$ copy each channel 4 times. Then, the generalized inverses $A_2^\dagger \colon \mathbb{R}^{32\times 32\times 4} \to \mathbb{R}^{32\times 32}$ and $A_3^\dagger \colon \mathbb{R}^{64\times 64\times 4} \to \mathbb{R}^{64\times 64}$ from (6) are given by taking the mean of the four channels of the input. The invertible neural networks $T_2 \colon \mathbb{R}^{32\times 32\times 4} \to \mathbb{R}^{64\times 64}$ and $T_3 \colon \mathbb{R}^{64\times 64\times 4} \to \mathbb{R}^{128\times 128}$ consist of 3 invertible blocks, where the subnetworks s_i and t_i, $i = 1, 2$ are convolutional neural networks with one hidden layer and 64 channels. After these three coupling blocks we

(a) Samples from the considered data set for the deblurring example.

(b) Learned chart with one generator. The figure shows the images $D(x)$ for 20 values of x equispaced in $[-1, 1]$.

(c) Learned charts with two generators. The figure shows the images $D_k(x)$ for 20 values of x equispaced in $[-1, 1]$ for $k = 1$ (top) and $k = 2$ (bottom).



(d) Reconstructions for the deblurring example. From top to bottom: ground truth image, observation, reconstruction with one generator and reconstruction with two generators.

Figure 7: Data set, learned charts and reconstructions for the deblurring example.

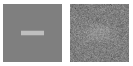
use an invertible upsampling (Etmann et al., 2020) to obtain the correct output dimension. For the normalizing flow in the latent space, we use an invertible neural network with three blocks, where the subnetworks s_i and t_i, $i = 1, 2$ are dense feed-forward networks with two hidden layers and 64 neurons.

We train all the models for 200 epochs with the Adam optimizer. Afterwards we apply the overlapping procedure for 50 epochs. See Algorithm 1 for the details of the training algorithm.

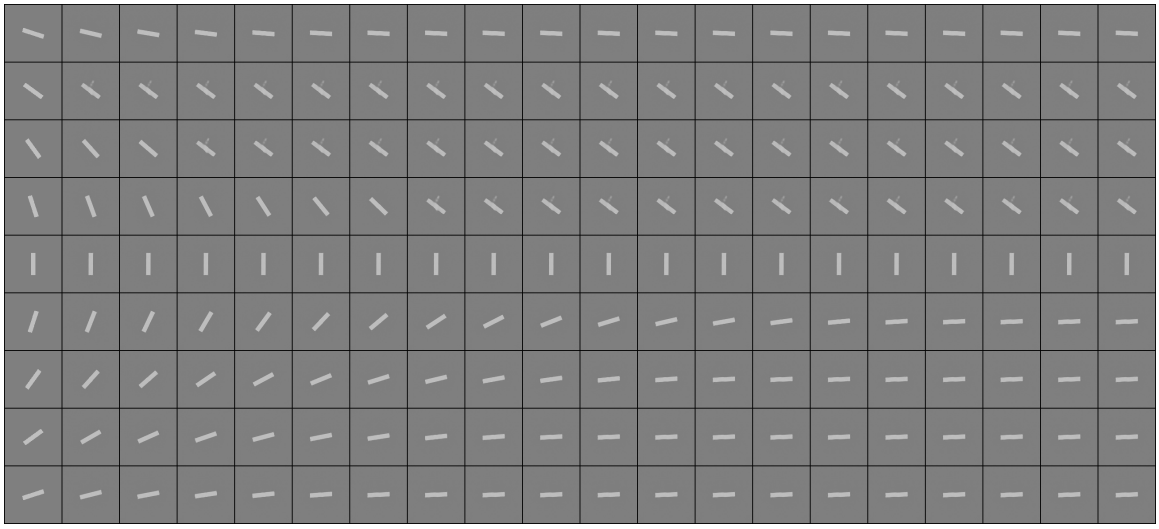
6.1 Deblurring

First, we consider the inverse problem of noisy image deblurring. Here, the forward operator $\mathcal{G}$ in (11) is linear and given by the convolution with a 30×30 Gaussian blur kernel with standard deviation 15. In order to obtain outputs y of the same size as the input x, we use constant padding with intensity $1/2$ within the convolution. Moreover, the image is corrupted by white Gaussian noise η with standard deviation 0.1. Given an observation y generated by this degradation process, we aim to reconstruct the unknown ground truth image x.

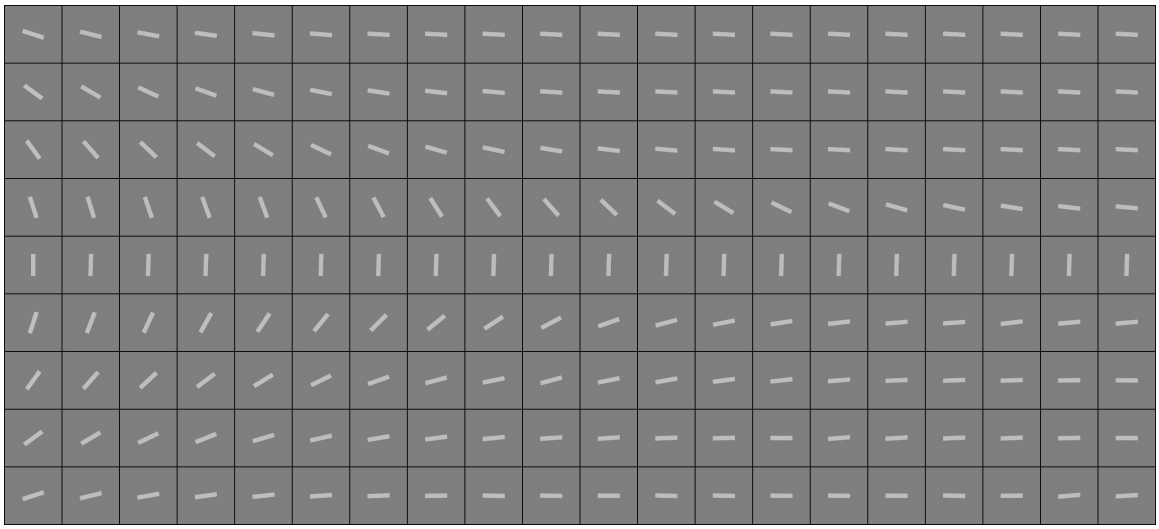
Data set and Manifold Approximation. Here, we consider the data set of 128×128 images showing a bright bar with a gray background that is centered and rotated. The intensity of fore- and background as well as the size of the bar are fixed. Some example images



(a) Ground truth (left) and observation (right).



(b) Visualization of the trajectories $(x_n)_n$ for different initializations x_0 with one generator. Left column: initialization, right column: reconstruction x_{250}, columns in between: images x_n for n approximately equispaced between 0 and 250.



(c) Visualization of the trajectories $(x_n)_n$ for different initializations x_0 with two generators. Left column: initialization, right column: reconstruction x_{250}, columns in between: images x_n for n approximately equispaced between 0 and 250.

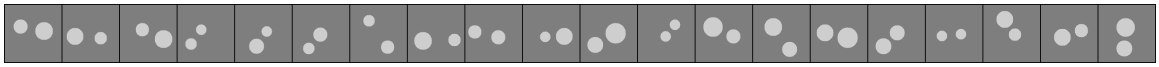
Figure 8: Gradient descent for the deblurring example.

from the data set are given in Figure 7a. The data set forms a one-dimensional manifold parameterized by the rotation of the bar. Therefore, it is homeomorphic to S^1 and does not admit a global parameterization since it contains a hole.

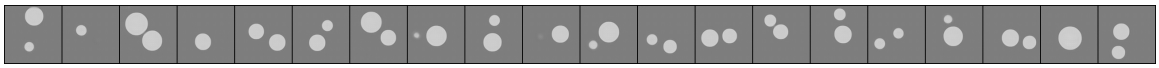
We approximate the data manifold by a mixture model of two VAEs and compare the result with the approximation with a single VAE, where the latent dimension is set to $d = 1$. The learned charts are visualized in Figure 7b and 7c. We observe that the charts learned with a mixture of two VAEs can generate all possible rotations and overlap at their boundaries. On the other hand, the chart learned with a single VAE does not cover all rotations but has a gap due to the injectivity of the decoder. This gap is also represented in the final test loss of the model, which approximates the negative log likelihood of the test data. It is given by 52.04 for one generator and by 39.84 for two generators. Consequently, the model with two generators fits the data manifold much better.

Reconstruction. In order to reconstruct the ground truth image, we use our gradient descent scheme for the function (12) as outlined in Algorithm 2 for 500 iterations. Since the function F is defined on the whole $\mathbb{R}^{128\times 128}$, we compute the Riemannian gradient $\nabla_{\mathcal{M}} F(x)$ accordingly to Remark 4. More precisely, for $x \in U_k$, we have $\nabla_{\mathcal{M}} F(x) = J(J^{\mathrm{T}}J)^{-1}J^{\mathrm{T}}\nabla F(x)$, where $\nabla F(x)$ is the Euclidean gradient of F and $J = \nabla \mathcal{D}_k(\mathcal{E}_k(x))$ is the Jacobian of the k-th decoder evaluated at $\mathcal{E}_k(x)$. Here, the Euclidean gradient $\nabla F(x)$ and the Jacobian matrix J are computed by algorithmic differentiation. Moreover, we use the retractions $\tilde{R}_{k,x}$ from (9). As initialization x_0 of our gradient descent scheme, we use a random sample from the mixture of VAEs. The results are visualized in Figure 7d. We observe that the reconstructions with two generators always recover the ground truth images very well. On the other hand, the reconstructions with one generator often are unrealistic and do not match with the ground truth. These unrealistic images appear at exactly those points where the chart of the VAE with one generator does not cover the data manifold.

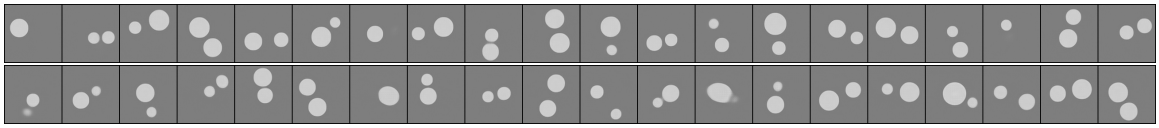
In order to better understand why the reconstructions with one generator often fail, we consider the trajectories $(x_n)_n$ generated by Algorithm 2 more in detail. We consider a fixed ground truth image showing a horizontal bar and a corresponding observation as given in Figure 8a. Then, we run Algorithm 2 for different initializations. The results are given in Figure 8b for one generator and in Figure 8c for two generators. The left column shows the initialization x_0, and in the right column, there are the values x_{250} after 250 gradient descent steps. The columns in between show the values x_n for (approximately) equispaced n between 0 and 250. With two generators, the trajectory $(x_n)_n$ are a smooth transition from the initialization to the ground truth. Only when the initialization is a vertical bar (middle row), the images x_n remain similar to the initialization x_0 for all n, since this is a critical point of the $F|_{\mathcal{M}}$ and hence the Riemannian gradient is zero. With one generator, we observe that some of the trajectories get stuck exactly at the gap, where the manifold is not covered by the chart. At this point the latent representation of the corresponding image would have to jump, which is not possible. Therefore, a second generator is required here.



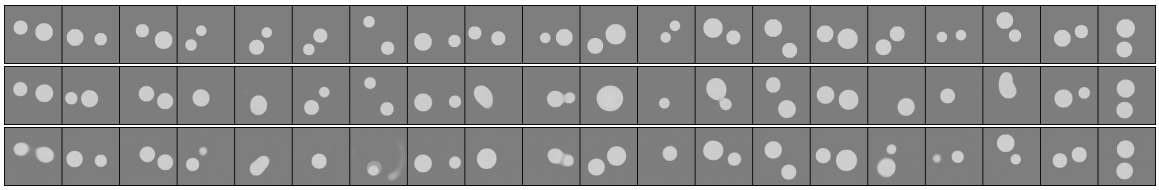
(a) Samples from the considered data set for the EIT example.



(b) Samples from the VAE with one generator.



(c) Samples from the mixture of two VAEs. Top: first chart, bottom: second chart.



(d) Reconstructions. From top to bottom: ground truth image, reconstruction with one generator, reconstruction with two generators.

Figure 9: Data set, synthesized samples and reconstructions for the EIT example.

6.2 Electrical Impedance Tomography

Finally, we consider the highly non-linear and ill-posed inverse problem of electrical impedance tomography (EIT, Cheney et al., 1999), which is also known in the mathematical literature as the Calderón problem (Astala and Päivärinta, 2006; Feldman et al., 2019; Mueller and Siltanen, 2012). EIT is a non-invasive, radiation-free method to measure the conductivity of a tissue through electrodes placed on the surface of the body. More precisely, electrical currents patterns are imposed on some of these electrodes and the resulting voltage differences are measured on the remaining ones. Although harmless, the use of this modality in practice is very limited because the standard reconstruction methods provide images with very low spatial resolution. This is an immediate consequence of the severe ill-posedness of the inverse problem (Alessandrini, 1988; Mandache, 2001).

Classical methods for solving this inverse problem include variational-type methods (Cheney et al., 1990), the Lagrangian method (Chen and Zou, 1999), the factorization method (Brühl and Hanke, 2000; Kirsch and Grinberg, 2008), the D-bar method (Siltanen et al., 2000), the enclosure method (Ikehata and Siltanen, 2000), and the monotonicity method (Tamburrino and Rubinacci, 2002). Similarly as many other inverse problems, deep learning methods have had a big impact on EIT. For example, Fan and Ying (2020) propose an end-to-end neural network that learns the forward map $\mathcal{G}$ and its inverse. Moreover, deep learning approaches can be combined with classical methods, e.g, by post processing methods (Hamilton and Hauptmann, 2018; Hamilton et al., 2019) or by variational learning algorithms (Seo et al., 2019).

Data set and Manifold Approximation. We consider the manifold consisting of 128×128 images showing two bright non-overlapping balls with a gray background, representing conductivities with special inclusions. The radius and the position of the balls vary, while the fore- and background intensities are fixed. Some exemplary samples of the data set are given in Figure 9a.

Remark 10 (Dimension and topology of the data manifold) *Since the balls are indistinguishable and not allowed to overlap, an image can be uniquely described by the angle between the two balls, the midpoint between both balls, their distance and the two radii. Hence, the data manifold is homeomorphic to* $\mathbb{S}^1 \times (0,1)^2 \times (0,1) \times (0,1)^2 = \mathbb{S}^1 \times (0,1)^5$. *In particular, it contains a hole and does not admit a global parameterization.*

A slightly more general version of this manifold was considered by Alberti et al. (2023), where Lipschitz stability is proven for a related inverse boundary value problem restricted to the manifold. Other types of inclusions (with unknown locations), notably polygonal and polyhedral inclusions, have been considered in the literature (Beretta et al., 2021; Beretta and Francini, 2022; Aspri et al., 2022). The case of small inclusions is discussed by Ammari and Kang (2004).

We approximate the data manifold by a mixture of two VAEs and compare the results with the approximation with a single VAE. The latent dimension is set to the manifold dimension, i.e., $d = 6$. Some samples of the learned charts are given in Figure 9b and 9c. As in the previous example, both models produce mostly realistic samples. The test loss is given by 365.21 for one generator and by 229.99 for two generators. Since the test loss approximates the negative log likelihood value of the test data, this indicates that two generators are needed in order to cover the whole data manifold.

The Forward Operator and its Derivative. From a mathematical viewpoint, EIT considers the following PDE with Neumann boundary conditions

$$\begin{cases} -\nabla \cdot (\gamma \, \nabla u_g) = 0 & \text{in } \Omega, \\ \gamma \, \partial_\nu u_g = g & \text{on } \partial\Omega, \end{cases} \tag{13}$$

where $\Omega \subseteq \mathbb{R}^2$ is a bounded domain, $\gamma \in L^\infty(\Omega)$ is such that $\gamma(x) \geq \lambda > 0$ and $u_g \in H^1(\Omega)$ is the unique weak solution with zero boundary mean of (13) with Neumann boundary data $g \in H_\diamond^{-\frac{1}{2}}(\partial\Omega)$, with $H_\diamond^s(\partial\Omega) = \{f \in H^s(\partial\Omega) : \int_{\partial\Omega} f ds = 0\}$. From the physical point of view, g represents the electric current applied on $\partial\Omega$ (through electrodes placed on the boundary of the body), u_g is the electric potential and γ is the conductivity of the body in the whole domain Ω. The inverse problem consists in the reconstruction of γ from the knowledge of all pairs of boundary measurements $(g, u_g|_{\partial\Omega})$, namely, of all injected currents together with the corresponding electric voltages generated at the boundary. In a compact form, the measurements may be modelled by the Neumann-to-Dirichlet map

$$\begin{aligned} \mathcal{G}(\gamma) \colon H_\diamond^{-\frac{1}{2}}(\partial\Omega) &\to H_\diamond^{\frac{1}{2}}(\partial\Omega) \\ g &\mapsto u_g\big|_{\partial\Omega}. \end{aligned}$$

Since the PDE (13) is linear, the map $\mathcal{G}(\gamma)$ is linear. However, the forward map $\gamma \mapsto \mathcal{G}(\gamma)$ is nonlinear in γ, and so is the corresponding inverse problem. The map $\mathcal{G}$ is continuously

differentiable, and its Fréchet derivative (see Harrach, 2019) is given by $[\nabla\mathcal{G}(\gamma)](\sigma)(g) = w_g|_{\partial\Omega}$, where $w_g \in H^1(\Omega)$ is the unique weak solution with zero boundary mean of

$$\begin{cases} -\nabla\cdot(\gamma\,\nabla w_g) = \nabla\cdot(\sigma\,\nabla u_g) & \text{in } \Omega, \\ -\gamma\,\partial_\nu w_g = \sigma\,\partial_\nu u_g & \text{on } \partial\Omega, \end{cases}$$

where $u_g \in H^1(\Omega)$ is the unique weak solution with zero boundary mean that solves (13). We included the expression of this derivative in the continuous setting for completeness, but, as a matter of fact, we will need only its semi-discrete counterpart given below.

Discretization and Objective Function. In our implementations, we discretize the linear mappings $G(\gamma)$ by restricting them to a finite dimensional subspace spanned by a-priori fixed boundary functions $g_1, \dots, g_N \in H_\diamond^{-\frac{1}{2}}(\partial\Omega)$. Then, following (Beretta et al., 2018, eqt. (2.2)), we reconstruct the conductivity by minimizing the semi-discrete functional

$$F(\gamma) = \frac{1}{2}\sum_{n=1}^{N}\int_{\partial\Omega}|u_{g_n}(s) - (u_{\text{true}})_{g_n}(s)|^2 ds, \tag{14}$$

where $(u_{\text{true}})_{g_n}$ is the observed data. In our discrete setting, we represent the conductivitiy γ by a piecewise constant function $\gamma = \sum_{m=1}^{M}\gamma_m \mathbb{1}_{T_m}$ on a triangulation $(T_m)_{m=1,\dots,M}$. Then, following Equation (2.20) from Beretta et al. (2018), the derivative of (14) with respect to γ is given by

$$\frac{dF}{d\gamma_m}(\gamma) = \sum_{n=1}^{N}\int_{T_m}\nabla u_{g_n}(x)\cdot\nabla z_{g_n}(x)dx, \tag{15}$$

where z_{g_n} solves

$$\begin{cases} -\nabla\cdot(\gamma\,\nabla z_{g_n}) = 0 & \text{in } \Omega, \\ \gamma\,\partial_\nu z_{g_n} = (u_{\text{true}})_{g_n} - u_{g_n} & \text{on } \partial\Omega, \end{cases} \tag{16}$$

with the normalization $\int_{\partial\Omega} z_{g_n}(s)ds = \int_{\partial\Omega}(u_{\text{true}})_{g_n}(s)ds$.

Implementation Details. In our experiments the domain Ω is given by the unit square $[0,1]^2$. For solving the PDEs (13) and (16), we use a finite element solver from the DOLFIN library (Logg and Wells, 2010). We employ meshes that are coarser in the middle of Ω and finer close to the boundary. To simulate the approximation error of the meshes, and to avoid inverse crimes, we use a fine mesh to generate the observation and a coarser one for the reconstructions. We use $N = 15$ boundary functions, which are chosen as follows. We divide each of the four edges of the unit square $[0,1]^2$ into 4 segments and denote by $b_1, \dots, b_{16}$ the functions that are equal to 1 on one of these segments and 0 otherwise. Then, we define the boundary functions as $g_n = \sum_{i=1}^{16} a_{n,i}b_i$, where the matrix $A = (a_{n,i})_{n=1,\dots,15,i=1,\dots,16}$ is the 16×16 Haar matrix without the first row. More precisely, the rows of A are given by the rows of the matrices $2^{-k/2}(\text{Id}_{2^{4-k}} \otimes (1,-1) \otimes e_{2^{k-1}}^{\mathrm{T}})$ for $k = 1, \dots, 4$, where $\otimes$ is the Kronecker product and $e_j \in \mathbb{R}^j$ is the vector where all entries are 1.

Results. We reconstruct the ground truth images from the observations by minimizing the functional F from (14) subject to $\gamma \in \mathcal{M}$. To this end, we apply the gradient descent scheme from Algorithm 2 for 100 steps. Since the evaluation of the forward operator and its derivative include the numerical solution of a PDE, it is computationally very costly. Hence, we aim to use as few iterations of Algorithm 2 as possible. To this end, we apply the adaptive step size scheme from Algorithm 3. As retractions we use $\tilde{R}_{k,x}$ from (9). The initialization γ_0 of the gradient descent scheme is given by a random sample from the mixture of VAEs.

Since F is defined on the whole $\mathbb{R}_+^{128\times 128}$, we use again Remark 4 for the evaluation of the Riemannian gradient. More precisely, for $\gamma \in U_k$, we have that $\nabla_{\mathcal{M}} F(\gamma) = J(J^{\mathrm{T}}J)^{-1}J^{\mathrm{T}}\nabla F(\gamma)$, where $\nabla F(\gamma)$ is the Euclidean gradient of F and $J = \nabla \mathcal{D}_k(\mathcal{E}_k(\gamma))$. Here, we compute $\nabla F(\gamma)$ by (15) and J by algorithmic differentiation.

The reconstructions for 20 different ground truths are visualized in Figure 9d. We observe that both models capture the ground truth structure in most cases, but also fail sometimes. Nevertheless, the reconstructions with the mixture of two VAEs recover the correct structure more often and more accurately than the single VAE, which can be explained by the better coverage of the data manifold. To quantify the difference more in detail, we rerun the experiment with 200 different ground truth images and compare the results with one and two generators using the PSNR and SSIM. As an additional evaluation metric, we run the segmentation algorithm proposed by Otsu (1979) on the ground truth and reconstruction image and compare the resulting segmentation masks using the SSIM. The resulting values are given in the following table.

	One generator	Two generators
PSNR	23.64 ± 3.91	24.76 ± 3.79
SSIM	0.8951 ± 0.0377	0.9111 ± 0.0368
segment+SSIM	0.8498 ± 0.0626	0.8667 ± 0.0614

Consequently, the reconstructions with two generators are significantly better than those with one generator for all evaluation metrics.

7. Conclusions

In this paper we introduced mixture models of VAEs for learning manifolds of arbitrary topology. The corresponding decoders and encoders of the VAEs provide analytic access to the resulting charts and are learned by a loss function that approximates the negative log-likelihood function. For minimizing functions F defined on the learned manifold we proposed a Riemannian gradient descent scheme. In the case of inverse problems, F is chosen as a data-fidelity term. Finally, we demonstrated the advantages of using several generators on numerical examples.

This work can be extended in several directions. First, gradient descent methods converge only locally and are not necessarily fast. Therefore, it would be interesting to extend the minimization of the functional F in Section 7 to higher-order methods or incorporate momentum parameters. Moreover, a careful choice of the initialization could improve the convergence behavior. Further, our reconstruction method could be extended to Bayesian inverse problems. Since the mixture model of VAEs provides us with a probability distribution and an (approximate) density, stochastic sampling methods like the Langevin

dynamics could be used for quantifying uncertainties within our reconstructions. Indeed, Langevin dynamics on Riemannian manifolds are still an active area of research. Further, for large numbers K of charts the mixture of VAEs might have a considerable number of parameters. As a remedy, we could incorporate the selection of the chart as conditioning parameter in one conditional decoder-encoder pair (see Sohn et al., 2015 as a reference for conditional VAEs). Finally, recent papers show that diffusion models provide an implicit representation of the data manifold (Stanczuk et al., 2024; Ross et al., 2024). It would be interesting to investigate optimization models on such manifolds in order to apply them to inverse problems.

Acknowledgments

This material is based upon work supported by the Air Force Office of Scientific Research under award numbers FA8655-20-1-7027 and FA8655-23-1-7083. We acknowledge the support of Fondazione Compagnia di San Paolo. Co-funded by the European Union (ERC, SAMPDE, 101041040). Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them. GSA, MS and SS are members of the "Gruppo Nazionale per l'Analisi Matematica, la Probabilità e le loro Applicazioni", of the "Istituto Nazionale di Alta Matematica". The research of GSA, MS and SS was supported in part by the MUR Excellence Department Project awarded to the Department of Mathematics, University of Genoa, CUP D33C23001110001. JH acknowledges funding by the German Research Foundation (DFG) within the project STE 571/16-1 and by the EPSRC programme grant "The Mathematics of Deep Learning" with reference EP/V026259/1.

Appendix A. Derivation of the ELBO

By Jensen's inequality the evidence can be lower-bounded by

$$\begin{aligned}
\log(p_{\tilde{X}}(x)) &= \log\Big(\int_{\mathbb{R}^d} p_{Z,\tilde{X}}(z,x)\,\mathrm{d}z\Big)\\
&= \log\Big(\int_{\mathbb{R}^d} \frac{p_{Z,\tilde{X}}(z,x)}{p_{\tilde{Z}|X=x}(z)} p_{\tilde{Z}|X=x}(z)\,\mathrm{d}z\Big)\\
&\geq \int_{\mathbb{R}^d} \log\Big(\frac{p_{Z,\tilde{X}}(z,x)}{p_{\tilde{Z}|X=x}(z)}\Big) p_{\tilde{Z}|X=x}(z)\,\mathrm{d}z\Big)\\
&= \mathbb{E}_{z\sim P_{\tilde{Z}|X=x}}\Big[\log\Big(\frac{p_Z(z)p_{\tilde{X}|Z=z}(x)}{p_{\tilde{Z}|X=x}(z)}\Big)\Big]\\
&= \mathbb{E}_{z\sim P_{\tilde{Z}|X=x}}[\log(p_Z(z)) + \log(p_{\tilde{X}|Z=z}(x)) - \log(p_{\tilde{Z}|X=x}(z))].
\end{aligned}$$

Accordingly to the definition of $\tilde{Z}$ and $\tilde{X}$, we have that $p_{\tilde{X}|Z=z}(x) = \mathcal{N}(x; D(z), \sigma_x^2 I_n)$ and $p_{\tilde{Z}|X=x}(z) = \mathcal{N}(z; E(x), \sigma_z^2 I_d)$. Thus, the above formula is, up to a constant, equal to

$$\mathbb{E}_{z\sim P_{\tilde{Z}|X=x}}[\log(p_Z(z)) - \tfrac{1}{2\sigma_x^2}\|x - D(z)\|^2 - \log(\mathcal{N}(z; E(x), \sigma_z^2 I_d))].$$

Considering the substitution $\xi = (z - E(x))/\sigma_z$, we obtain

$$\mathbb{E}_{\xi\sim\mathcal{N}(0,I_d)}[\log(p_Z(E(x)+\sigma_z\xi)) - \tfrac{1}{2\sigma_x^2}\|D(E(x)+\sigma_z\xi) - x\|^2 - \log(\mathcal{N}(\xi;0,I_d))].$$

Note that also the last summand does not depend on D and E. Thus, we obtain, up to a constant, the *evidence lower bound* (ELBO) given by

$$\mathrm{ELBO}(x|\theta) \coloneqq \mathbb{E}_{\xi\sim\mathcal{N}(0,I_d)}[\log(p_Z(E(x)+\sigma_z\xi)) - \tfrac{1}{2\sigma_x^2}\|D(E(x)+\sigma_z\xi) - x\|^2].$$

Appendix B. Error Bound for the Approximation of β_{ik} by $\tilde{\beta}_{ik}$

It is well-known that the difference between evidence and ELBO can be expressed in terms of the Kullback-Leibler divergence (see Kingma and Welling, 2019, Sec 2.2). In the special case that $E \circ D = \mathrm{Id}$, which is relevant in this paper, this estimate can be simplified by the following lemma. To this end, denote by

$$\mathcal{F}(x|\theta) \coloneqq \mathbb{E}_{z\sim P_{\tilde{Z}|X=x}}[\log(p_Z(z)) + \log(p_{\tilde{X}|Z=z}(x)) - \log(p_{\tilde{Z}|X=x}(z))] = \mathrm{ELBO}(x|\theta) + \mathrm{const}$$

the ELBO before leaving out the constants.

Lemma 11 *Assume that $E \circ D = \mathrm{Id}$. Then, it holds*

$$\log(p_{\tilde{X}}(x)) - \mathcal{F}(x|\theta) = \mathrm{KL}(\mathcal{N}(E(x),\sigma_z^2 I_d), E_\#\mathcal{N}(x,\sigma_x^2 I_n)).$$

Proof By Equation (2.8) in Kingma and Welling (2019), it holds that

$$\log(p_{\tilde{X}}(x)) - \mathcal{F}(x|\theta) = \mathrm{KL}(P_{\tilde{Z}|X=x}, P_{Z|\tilde{X}=x}).$$

Inserting the definitions $\tilde{Z} = X + \xi$ and $Z = E(D(Z)) = E(D(Z) + \eta - \eta) = E(\tilde{X} - \eta)$, this is equal to

$$\mathrm{KL}(P_{E(X)+\xi|X=x}, P_{E(\tilde{X}-\eta)|\tilde{X}=x}) = \mathrm{KL}(P_{E(x)+\xi}, P_{E(x-\eta)}) = \mathrm{KL}(P_{E(x)+\xi}, E_\# P_{x-\eta}).$$

Using $\xi \sim \mathcal{N}(0,\sigma_z^2 I_d)$ and $\eta = \mathcal{N}(0,\sigma_x^2 I_n)$, we arrive at the assertion. ■

The next two lemmas exploit this estimate for bounding the approximation error between $\tilde{\beta}_{ik}$ and β_{ik}.

Lemma 12 *Assume that* $\exp(\mathrm{KL}(\mathcal{N}(E_k(x_i),\sigma_z^2 I_d), E_{k\#}\mathcal{N}(x_i,\sigma_x^2 I_n))) \le L$ *for all* x_i, k. *Then*

$$\frac{1}{L}\beta_{ik} \le \tilde{\beta}_{ik} \le L\beta_{ik},$$

where β_{ik} *and* $\tilde{\beta}_{ik}$ *are defined in* (2) *and* (3).

Proof Due to Lemma 11, we have that

$$1 \le \frac{p_{\tilde{X}_k}(x_i)}{\exp(\mathcal{F}(x_i|\theta_k))} \le \exp(\mathrm{KL}(\mathcal{N}(E_k(x_i),\sigma_z^2 I_d), E_{k\#}\mathcal{N}(x_i,\sigma_x^2 I_n))) \le L,$$

i.e.,

$$\frac{1}{L} p_{\tilde{X}_k}(x_i) \leq \exp(\mathcal{F}(x_i|\theta_k)) \leq p_{\tilde{X}_k}(x_i).$$

Since by $\mathrm{ELBO}(x_i|\theta_k) = \mathcal{F}(x_i|\theta_k) + \text{const}$, it holds that $\tilde{\beta}_{ik} = \frac{\alpha_k \exp(\mathcal{F}(x_i|\theta_k))}{\sum_{j=1}^K \alpha_j \exp(\mathcal{F}(x_i|\theta_j))}$, this implies that

$$\frac{1}{L}\beta_{ik} = \frac{\frac{1}{L}\alpha_k p_{\tilde{X}_k}(x_i)}{\sum_{j=1}^K \alpha_j p_{\tilde{X}_k}(x_i)} \leq \underbrace{\frac{\alpha_k \exp(\mathcal{F}(x_i|\theta_k))}{\sum_{j=1}^K \alpha_j \exp(\mathcal{F}(x_i|\theta_j))}}_{=\tilde{\beta}_{ik}} \leq \frac{\alpha_k p_{\tilde{X}_k}(x_i)}{\frac{1}{L}\sum_{j=1}^K \alpha_j p_{\tilde{X}_k}(x_i)} = L\beta_{ik},$$

which concludes the proof. ■

Lemma 13 *Let $E = A \circ T$ where $A\colon \mathbb{R}^n \to \mathbb{R}^d$ is given by the matrix $A = (I_d|0)$ and T is invertible and bi-Lipschitz with Lipschitz constants L_1 and L_2 for T and T^{-1}. Then*

$$\mathrm{KL}(\mathcal{N}(E(x), \sigma_z^2 I_d), E_\# \mathcal{N}(x, \sigma_x^2 I_n)) \leq \log(L_1^n L_2^n) + \frac{d}{2}\left(\frac{L_2^2\sigma_z^2}{\sigma_x^2} - 1 - \log\left(\frac{L_2^2\sigma_z^2}{\sigma_x^2}\right)\right).$$

Proof We estimate the density $p_{E_\#\mathcal{N}(x,\sigma_x^2 I_n))}(z)$ from below. By (Altekrüger et al., 2023, Lemma 4), we have that

$$p_{T_\#\mathcal{N}(x,\sigma_x^2 I_n)} \geq \frac{1}{L_1^n L_2^n}\mathcal{N}(T(x), \frac{\sigma_x}{L_2^2} I_n).$$

Using the projection property of Gaussian distributions, this implies that $E_\#\mathcal{N}(x, \sigma_x^2 I_n)) = A_\#(T_\#\mathcal{N}(x, \sigma_x^2 I_n))$ fulfills

$$p_{E_\#\mathcal{N}(x,\sigma_x^2 I_n))} \geq \frac{1}{L_1^n L_2^n}\mathcal{N}(E(x), \frac{\sigma_x}{L_2^2} I_d).$$

Hence, by the definition of the Kullback-Leibler divergence, it holds

$$\begin{aligned}
&\mathrm{KL}(\mathcal{N}(E(x), \sigma_z^2 I_d), E_\#\mathcal{N}(x, \sigma_x^2 I_n)) \\
&= \mathbb{E}_{z\sim\mathcal{N}(E(x),\sigma_z^2 I_d)}\Big[\log\Big(\frac{\mathcal{N}(z|E(x), \sigma_z^2 I_d)}{p_{E_\#\mathcal{N}(x,\sigma_x^2 I_n))}(z)}\Big)\Big] \\
&\leq \log(L_1^n L_2^n) + \mathbb{E}_{z\sim\mathcal{N}(E(x),\sigma_z^2 I_d)}\Big[\log\Big(\frac{\mathcal{N}(z|E(x), \sigma_z^2 I_d)}{\mathcal{N}(z|E(x), \frac{\sigma_x^2}{L_2^2} I_d)}\Big)\Big] \\
&= \log(L_1^n L_2^n) + \mathrm{KL}(\mathcal{N}(E(x), \sigma_z^2 I_d), \mathcal{N}(z|E(x), \frac{\sigma_x^2}{L_2^2} I_d)).
\end{aligned}$$

Inserting the formula for the KL divergence between two normal distributions, we obtain

$$\begin{aligned}
\mathrm{KL}(\mathcal{N}(E(x), \sigma_z^2 I_d), \mathcal{N}(z|E(x), \frac{\sigma_x^2}{L_2^2} I_d)) &= \frac{1}{2}\big(\mathrm{trace}(\tfrac{L_2^2\sigma_z^2}{\sigma_x^2} I_d) - d + d\log(\tfrac{\sigma_x^2}{L_2^2\sigma_z^2})\big) \\
&= \frac{d}{2}\left(\frac{L_2^2\sigma_z^2}{\sigma_x^2} - 1 - \log\left(\frac{L_2^2\sigma_z^2}{\sigma_x^2}\right)\right),
\end{aligned}$$

which proves the claim. ■

Combining the previous two lemmas, we obtain the following corollary.

Corollary 14 *Assume that $E_1, ..., E_K$ are of the form $E_k = A \circ T_k$, where $A\colon \mathbb{R}^n \to \mathbb{R}^d$ is given by the matrix $A = (I_d|0)$ and T_k is invertible and bi-Lipschitz with Lipschitz constants L_1 and L_2 for T and T^{-1} and assume that $E_k \circ D_k = \mathrm{Id}$. Then*

$$\frac{1}{L}\beta_{ik} \leq \tilde{\beta}_{ik} \leq L\beta_{ik},$$

where β_{ik} and $\tilde{\beta}_{ik}$ are given in (2) and (3) and $L = L_1^n L_2^n \exp(\frac{d}{2}(\frac{L_2^2\sigma_z^2}{\sigma_x^2} - 1 - \log(\frac{L_2^2\sigma_z^2}{\sigma_x^2})))$.

Note in particular that $\tilde{\beta}_{ik} \to 0$ whenever $\beta_{ik} \to 0$ and, by

$$\tilde{\beta}_{ik} = 1 - \sum_{\substack{l=1 \\ l\neq k}}^{K} \tilde{\beta}_{ik} \geq 1 - L\sum_{\substack{l=1 \\ l\neq k}}^{K} \beta_{ik} = 1 - L(1-\beta_{ik}) = 1 - L + L\beta_{ik}$$

that $\tilde{\beta}_{ik} \to 1$ whenever $\beta_{ik} \to 1$. However, the constant L from the corollary depends exponentially on the dimension n, which might limit its applicability when n is very large.

References

P.-A. Absil and J. Malick. Projection-like retractions on matrix manifolds. *SIAM Journal on Optimization*, 22(1):135–158, 2012.

P.-A. Absil, R. Mahony, and R. Sepulchre. Optimization algorithms on matrix manifolds. In *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2009.

G. S. Alberti, A. Arroyo, and M. Santacesaria. Inverse problems on low-dimensional manifolds. *Nonlinearity*, 36(1):734–808, 2023.

G. S. Alberti, M. Santacesaria, and S. Sciutto. Continuous Generative Neural Networks: A Wavelet-Based Architecture in Function Spaces. *Numerical Functional Analysis and Optimization*, to appear.

G. Alessandrini. Stable determination of conductivity by boundary measurements. *Applicable Analysis*, 27(1-3):153–172, 1988.

F. Altekrüger, A. Denker, P. Hagemann, J. Hertrich, P. Maass, and G. Steidl. PatchNR: Learning from very few images by patch normalizing flow regularization. *Inverse Problems*, 39(6):064006, 2023.

H. Ammari and H. Kang. *Reconstruction of small inhomogeneities from boundary measurements*, volume 1846 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 2004. ISBN 3-540-22483-1.

L. Ardizzone, J. Kruse, C. Rother, and U. Köthe. Analyzing inverse problems with invertible neural networks. In *International Conference on Learning Representations*, 2019.

S. Arridge, P. Maass, O. Öktem, and C.-B. Schönlieb. Solving inverse problems using data-driven models. *Acta Numerica*, 28:1–174, 2019.

M. Asim, F. Shamshad, and A. Ahmed. Blind image deconvolution using deep generative priors. *IEEE Transactions on Computational Imaging*, 6:1493–1506, 2020.

A. Aspri, E. Beretta, E. Francini, and S. Vessella. Lipschitz stable determination of polyhedral conductivity inclusions from local boundary measurements. *SIAM Journal on Mathematical Analysis*, 54(5):5182–5222, 2022.

K. Astala and L. Päivärinta. Calderón's inverse conductivity problem in the plane. *Annals of Mathematics*, 163(1):265–299, 2006.

E. Banijamali, A. Ghodsi, and P. Popuart. Generative mixture of networks. In *2017 International Joint Conference on Neural Networks*, pages 3753–3760. IEEE, 2017.

J. Behrmann, W. Grathwohl, R. T. Chen, D. Duvenaud, and J.-H. Jacobsen. Invertible residual networks. In *International Conference on Machine Learning*, pages 573–582. PMLR, 2019.

Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8): 1798–1828, 2013.

E. Beretta and E. Francini. Global Lipschitz stability estimates for polygonal conductivity inclusions from boundary measurements. *Applicable Analysis*, 101(10):3536–3549, 2022.

E. Beretta, S. Micheletti, S. Perotto, and M. Santacesaria. Reconstruction of a piecewise constant conductivity on a polygonal partition via shape optimization in EIT. *Journal of Computational Physics*, 353:264–280, 2018.

E. Beretta, E. Francini, and S. Vessella. Lipschitz stable determination of polygonal conductivity inclusions in a two-dimensional layered medium from the Dirichlet-to-Neumann map. *SIAM Journal on Mathematical Analysis*, 53(4):4303–4327, 2021.

A. Bora, A. Jalal, E. Price, and A. G. Dimakis. Compressed sensing using generative models. In *International Conference on Machine Learning*, pages 537–546. PMLR, 2017.

M. Brand. Charting a manifold. *Advances in Neural Information Processing Systems*, 15, 2002.

M. Brühl and M. Hanke. Numerical implementation of two noniterative methods for locating inclusions by impedance tomography. *Inverse Problems*, 16(4):1029, 2000.

F. Camastra and A. Staiano. Intrinsic dimension estimation: Advances and open problems. *Information Sciences*, 328:26–41, 2016.

N. Chen, A. Klushyn, F. Ferroni, J. Bayer, and P. Van Der Smagt. Learning flat latent manifolds with VAEs. In *International Conference on Machine Learning*, pages 1587–1596. PMLR, 2020.

R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud. Neural ordinary differential equations. *Advances in Neural Information Processing Systems*, 31, 2018.

R. T. Chen, J. Behrmann, D. K. Duvenaud, and J.-H. Jacobsen. Residual flows for invertible generative modeling. *Advances in Neural Information Processing Systems*, 32, 2019.

Z. Chen and J. Zou. An augmented Lagrangian method for identifying discontinuous parameters in elliptic systems. *SIAM Journal on Control and Optimization*, 37(3):892–910, 1999.

M. Cheney, D. Isaacson, J. C. Newell, S. Simske, and J. Goble. NOSER: An algorithm for solving the inverse conductivity problem. *International Journal of Imaging systems and technology*, 2(2):66–75, 1990.

M. Cheney, D. Isaacson, and J. C. Newell. Electrical impedance tomography. *SIAM Review*, 41(1):85–101, 1999.

T. Cohn, N. Devraj, and O. C. Jenkins. Topologically-informed atlas learning. *arXiv preprint arXiv:2110.00429*, 2021.

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

T. R. Davidson, L. Falorsi, N. De Cao, T. Kipf, and J. M. Tomczak. Hyperspherical variational auto-encoders. In *Uncertainty in Artificial Intelligence*, pages 856–865, 2018.

L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using real NVP. In *International Conference on Learning Representations*, 2016.

M. Duff, N. D. Campbell, and M. J. Ehrhardt. Regularising inverse problems with generative machine learning models. *arXiv preprint arXiv:2107.11191*, 2021.

C. Etmann, R. Ke, and C.-B. Schönlieb. iUNets: learnable invertible up-and downsampling for large-scale inverse problems. In *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing*, pages 1–6. IEEE, 2020.

F. Falck, H. Zhang, M. Willetts, G. Nicholson, C. Yau, and C. C. Holmes. Multi-facet clustering variational autoencoders. *Advances in Neural Information Processing Systems*, 34:8676–8690, 2021.

M. Fan, N. Gu, H. Qiao, and B. Zhang. Intrinsic dimension estimation of data by principal component analysis. *arXiv preprint arXiv:1002.2050*, 2010.

Y. Fan and L. Ying. Solving electrical impedance tomography with deep learning. *Journal of Computational Physics*, 404:109119, 2020.

J. Feldman, M. Salo, and G. Uhlmann. The Calderón problem—an introduction to inverse problems. *Preliminary notes on the book in preparation*, 2019.

D. Floryan and M. D. Graham. Data-driven discovery of intrinsic dynamics. *Nature Machine Intelligence*, 4(12):1113–1120, 2022.

M. González, A. Almansa, and P. Tan. Solving inverse problems by joint posterior maximization with autoencoding prior. *SIAM Journal on Imaging Sciences*, 15(2):822–859, 2022.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27, pages 2672–2680, 2014.

A. Goujon, S. Neumayer, P. Bohra, S. Ducotterd, and M. Unser. A neural-network-based convex regularizer for inverse problems. *IEEE Transactions on Computational Imaging*, 2023.

W. Grathwohl, R. T. Chen, J. Bettencourt, I. Sutskever, and D. Duvenaud. FFJORD: Free-form continuous dynamics for scalable reversible generative models. In *International Conference on Learning Representations*, 2018.

P. Hagemann, J. Hertrich, and G. Steidl. Stochastic normalizing flows for inverse problems: a Markov chains viewpoint. *SIAM/ASA Journal on Uncertainty Quantification*, 10(3): 1162–1190, 2022.

P. Hagemann, J. Hertrich, and G. Steidl. *Generalized normalizing flows via Markov chains*. Cambridge University Press, 2023.

S. J. Hamilton and A. Hauptmann. Deep D-bar: Real-time electrical impedance tomography imaging with deep neural networks. *IEEE Transactions on Medical Imaging*, 37(10):2367–2377, 2018.

S. J. Hamilton, A. Hänninen, A. Hauptmann, and V. Kolehmainen. Beltrami-net: domain-independent deep D-bar learning for absolute imaging with electrical impedance tomography (a-EIT). *Physiological Measurement*, 40(7):074002, 2019.

B. Harrach. Uniqueness and lipschitz stability in electrical impedance tomography with finitely many electrodes. *Inverse Problems*, 35(2):024005, 2019.

J. Hertrich. Proximal residual flows for Bayesian inverse problems. In *International Conference on Scale Space and Variational Methods in Computer Vision*, pages 210–222. Springer, 2023.

J. Hertrich, A. Houdard, and C. Redenbach. Wasserstein patch prior for image superresolution. *IEEE Transactions on Computational Imaging*, 8:693–704, 2022.

J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Q. Hoang, T. D. Nguyen, T. Le, and D. Phung. MGAN: Training generative adversarial nets with multiple generators. In *International Conference on Learning Representations*, 2018.

C.-W. Huang, D. Krueger, A. Lacoste, and A. Courville. Neural autoregressive flows. In *International Conference on Machine Learning*, pages 2078–2087. PMLR, 2018.

C. M. Hyun, S. H. Baek, M. Lee, S. M. Lee, and J. K. Seo. Deep learning-based solvability of underdetermined inverse problems in medical imaging. *Medical Image Analysis*, 69: 101967, 2021.

M. Ikehata and S. Siltanen. Numerical method for finding the convex hull of an inclusion in conductivity from boundary measurements. *Inverse Problems*, 16(4):1043, 2000.

A. J. Izenman. Introduction to manifold learning. *WIREs Computational Statistics*, 4(5): 439–446, 2012.

Z. Jiang, Y. Zheng, H. Tan, B. Tang, and H. Zhou. Variational deep embedding: An unsupervised and generative approach to clustering. In *International Joint Conference on Artificial Intelligence*, 2017.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

D. P. Kingma and P. Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in Neural Information Processing Systems*, 31, 2018.

D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.

D. P. Kingma and M. Welling. An introduction to variational autoencoders. *Foundations and Trends in Machine Learning*, 12(4):307–392, 2019.

A. Kirsch and N. Grinberg. *The factorization method for inverse problems*, volume 36 of *Oxford Lecture Series in Mathematics and its Applications*. Oxford University Press, 2008.

B. Kivva, G. Rajendran, P. Ravikumar, and B. Aragam. Identifiability of deep generative models without auxiliary information. *Advances in Neural Information Processing Systems*, 35:15687–15701, 2022.

E. O. Korman. Autoencoding topology. *arXiv preprint arXiv:1803.00156*, 2018.

E. O. Korman. Atlas based representation and metric learning on manifolds. *arXiv preprint arXiv:2106.07062*, 2021.

K. Kothari, A. Khorashadizadeh, M. de Hoop, and I. Dokmanić. Trumpets: Injective flows for inference and inverse problems. In *Uncertainty in Artificial Intelligence*, pages 1269–1278. PMLR, 2021.

E. Levina and P. Bickel. Maximum likelihood estimation of intrinsic dimension. *Advances in Neural Information Processing Systems*, 17, 2004.

F. Locatello, D. Vincent, I. Tolstikhin, G. Rätsch, S. Gelly, and B. Schölkopf. Competitive training of mixtures of independent deep generative models. *arXiv preprint arXiv:1804.11130*, 2018.

A. Logg and G. N. Wells. DOLFIN: Automated finite element computing. *ACM Transactions on Mathematical Software*, 37(2):1–28, 2010.

S. Lunz, O. Öktem, and C.-B. Schönlieb. Adversarial regularizers in inverse problems. *Advances in Neural Information Processing Systems*, 31, 2018.

Y. Ma and Y. Fu. *Manifold learning theory and applications*. CRC press, 2011.

N. Mandache. Exponential instability in an inverse problem for the Schrödinger equation. *Inverse Problems*, 17(5):1435, 2001.

P. Massa, S. Garbarino, and F. Benvenuto. Approximation of discontinuous inverse operators with neural networks. *Inverse Problems*, 38(10):Paper No. 105001, 16, 2022.

E. Mathieu, C. Le Lan, C. J. Maddison, R. Tomioka, and Y. W. Teh. Continuous hierarchical representations with Poincaré variational auto-encoders. *Advances in Neural Information Processing Systems*, 32, 2019.

J. L. Mueller and S. Siltanen. *Linear and nonlinear inverse problems with practical applications*. SIAM, 2012.

G. Ongie, A. Jalal, C. A. Metzler, R. G. Baraniuk, A. G. Dimakis, and R. Willett. Deep learning techniques for inverse problems in imaging. *IEEE Journal on Selected Areas in Information Theory*, 1(1):39–56, 2020.

D. Onken, S. W. Fung, X. Li, and L. Ruthotto. OT-flow: Fast and accurate continuous normalizing flows via optimal transport. In *AAAI Conference on Artificial Intelligence*, volume 35, pages 9223–9232, 2021.

N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

K. Pearson. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

E. Pineau and M. Lelarge. InfoCatVAE: representation learning with categorical variational autoencoders. *arXiv preprint arXiv:1806.08240*, 2018.

N. Pitelis, C. Russell, and L. Agapito. Learning a manifold as an atlas. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1642–1649, 2013.

L. A. P. Rey, V. Menkovski, and J. W. Portegies. Diffusion variational autoencoders. In *International Joint Conference on Artificial Intelligence*, pages 2704–2710, 2020.

D. Rezende and S. Mohamed. Variational inference with normalizing flows. In *International Conference on Machine Learning*, pages 1530–1538. PMLR, 2015.

B. L. Ross, G. Loaiza-Ganem, A. L. Caterini, and J. C. Cresswell. Neural implicit manifold learning for topology-aware density estimation. *Transactions on Machine Learning Research*, 2024.

S. Schonsheck, J. Chen, and R. Lai. Chart auto-encoders for manifold structured data. *arXiv preprint arXiv:1912.10094*, 2019.

J. K. Seo, K. C. Kim, A. Jargal, K. Lee, and B. Harrach. A learning-based method for solving ill-posed nonlinear inverse problems: a simulation study of lung EIT. *SIAM Journal on Imaging Sciences*, 12(3):1275–1295, 2019.

S. Sidheekh, C. B. Dock, T. Jain, R. Balan, and M. K. Singh. VQ-Flows: Vector quantized local normalizing flows. In *Uncertainty in Artificial Intelligence*, pages 1835–1845. PMLR, 2022.

S. Siltanen, J. Mueller, and D. Isaacson. An implementation of the reconstruction algorithm of A Nachman for the 2d inverse conductivity problem. *Inverse Problems*, 16(3):681, 2000.

K. Sohn, H. Lee, and X. Yan. Learning structured output representation using deep conditional generative models. *Advances in Neural Information Processing Systems*, 28, 2015.

Y. Song and S. Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

J. P. Stanczuk, G. Batzolis, T. Deveney, and C.-B. Schönlieb. Diffusion models encode the intrinsic dimension of data manifolds. In *International Conference on Machine Learning*, 2024.

J. Stolberg-Larsen and S. Sommer. Atlas generative models and geodesic interpolation. *Image and Vision Computing*, page 104433, 2022.

A. Tamburrino and G. Rubinacci. A new non-iterative inversion method for electrical resistance tomography. *Inverse Problems*, 18(6):1809, 2002.

F. Ye and A. G. Bors. Deep mixture generative autoencoders. *IEEE Transactions on Neural Networks and Learning Systems*, 33(10):5789–5803, 2022.