Disentangled Representation Learning and Generation with Manifold Optimization

Arun Pandey[1], **Michaël Fanuel**[2], **Joachim Schreurs**[1], **Johan A.K. Suykens**[1]

[1]KU Leuven, Department of Electrical Engineering (ESAT),

STADIUS Center for Dynamical Systems, Signal Processing and Data Analytics,

Kasteelpark Arenberg 10, B-3001 Leuven, Belgium.

[2]Université de Lille, CNRS, Centrale Lille,

UMR 9189 – CRIStAL, F-59000 Lille, France.

Keywords: Generative models, Latent variable models, Disentanglement

Abstract

Disentanglement is a useful property in representation learning which increases the interpretability of generative models such as Variational autoencoders (VAE), Generative Adversarial Models, and their many variants. Typically in such models, an increase in disentanglement performance is traded-off with generation quality.

In the context of latent space models, this work presents a representation learning framework that explicitly promotes disentanglement by encouraging orthogonal directions of variations. The proposed objective is the sum of an autoencoder error term along with a Principal Component Analysis reconstruction error in the feature space. This has an interpretation of a Restricted Kernel Machine with the eigenvector matrix valued on the Stiefel manifold. Our analysis shows that such a construction promotes disentanglement by matching the principal directions in the latent space with the directions of orthogonal variation in data space. In an alternating minimization scheme, we use Cayley ADAM algorithm — a stochastic optimization method on the Stiefel manifold along with the ADAM optimizer. Our theoretical discussion and various experiments show that the proposed model improves over many VAE variants in terms of both generation quality and disentangled representation learning.

1 Introduction

Latent space models are popular tools for sampling from high-dimensional distributions. Often, only a small number of latent factors are sufficient to describe data variations. These models exploit the underlying structure of the data and learn explicit representations that are faithful to the data generating factors. Popular latent space models are Variational autoencoders (VAEs) (Kingma & Welling, 2014), Restricted Boltzmann Machines (RBMs) (Salakhutdinov & Hinton, 2009), Normalizing Flows (Rezende & Mohamed, 2015), and their many variants.

In latent variable models, one is often interested in modeling the data in terms of *uncorrelated* or *independent* components, yielding a so-called 'disentangled' representation (Bengio et al., 2013) which is often studied in the context of VAEs. Generative Adversarial Networks (GAN) have also been extended to perform disentangled representation learning, for instance, with Info-GANs. It is a GAN that also maximizes the mutual information between a small subset of the discrete latent codes and the true images. In principle, disentanglement corresponds to identifying the underlying factors which generate the data. Components corresponding to the orthogonal directions in latent space may be interpreted as generating distinct factors in the input space e.g. lighting conditions, style, colors, etc. An illustration of a latent traversal is shown in Figure 1, where one observes that only one specific feature of the image is changing as one moves along a component in the latent space. For instance, in Figure 1, we observe that moving along the first component (vector $\boldsymbol{u}_1$) generates images where only floor color is varying while all other features such as shape, scale, wall color, object color, etc. are constant. Whereas traversing along the sixth component (vector $\boldsymbol{u}_6$) for instance, generates images where only the object scale changes as shown in the second row. As we explain later, the components here refer to the principal components given by the Principal Component Analysis (PCA). Therefore these principal directions encode the directions of maximum variance. Since the floor color is encoded by the largest number of pixels, it gets represented by the first principal component $\boldsymbol{u}_1$. Similarly, the other components correspond to the directions with smaller variance. An advantage of such a representation is that the different latent units impart more interpretability to the model. Disentangled models are useful for the generation of plausible pseudo-data with certain desirable properties, e.g. generating new car designs with a predefined color or height.

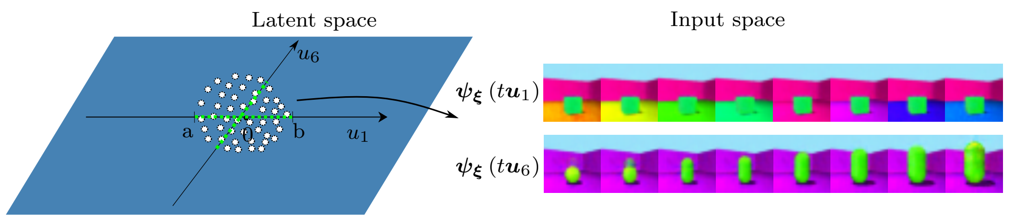


Figure 1: Images by the decoder of the latent space traversal, i.e., $\boldsymbol{\psi}_{\boldsymbol{\xi}}(t\boldsymbol{u}_i)$ for $t \in [a, b]$ with $a < b$ and for some $i \in \{1, \ldots, m\}$. Green and black dashed lines represent the walk along $\boldsymbol{u}_1$ and $\boldsymbol{u}_6$ respectively. At every step of the walk, the output of the decoder generates the data in the input space. The images were generated by St-RKM with $\sigma = 10^{-3}$ on 3Dshapes dataset; see Figure 5 for traversal along other components.

Now we introduce the mathematical setting to formalize our discussion throughout the paper. We start by introducing a VAE (Kingma & Welling, 2014). Let $p(\boldsymbol{x})$ be the distribution of the data $\boldsymbol{x} \in \mathbb{R}^d$ and consider latent vectors $\boldsymbol{z} \in \mathbb{R}^\ell$ with the prior distribution $p(\boldsymbol{z})$, typically a standard normal distribution. Then, one defines an encoder $q(\boldsymbol{z}|\boldsymbol{x})$ that can be deterministic or probabilistic, for e.g. given by $\mathcal{N}(\boldsymbol{z}|\boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}), \gamma^2\mathbb{I})$, where the mean[1] is given by the neural network $\boldsymbol{\phi}_{\boldsymbol{\theta}}$ parametrized by $\boldsymbol{\theta}$. A random decoder $p(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\psi}_{\boldsymbol{\xi}}(\boldsymbol{z}), \sigma_0^2\mathbb{I})$ is associated to the decoder neural network $\boldsymbol{\psi}_{\boldsymbol{\xi}}$, parametrized by $\boldsymbol{\xi}$, which maps latent codes to the data points. A VAE is trained by maximizing the lower bound to the idealized

[1] A typical implementation of VAE includes another neural network (after the primary network) for parametrizing the covariance matrix. To simplify this introductory discussion, this matrix is here chosen as a constant diagonal $\gamma^2\mathbb{I}$.

log-likelihood as below:

$$\mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z}|\boldsymbol{x})}[\log(p(\boldsymbol{x}|\boldsymbol{z}))] - \beta\,\mathrm{KL}(q(\boldsymbol{z}|\boldsymbol{x}), p(\boldsymbol{z})) \leq \log p(\boldsymbol{x}). \tag{1}$$

This lower bound is often called as the Evidence Lower Bound (ELBO) when $\beta = 1$. Higgins et al. (2017) show that the larger values of $\beta > 1$ promote more disentanglement but at the expense of generation quality. In this paper, we attempt to reconcile the generation quality with disentanglement. To introduce the model, we first make explicit the connection between β-VAEs and standard autoencoders (AEs). Let the dataset be $\{\boldsymbol{x}_i\}_{i=1}^n$ with $\boldsymbol{x}_i \in \mathbb{R}^d$. Let $q(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}|\boldsymbol{\phi_\theta}(\boldsymbol{x}), \gamma^2\mathbb{I})$ be an encoder, where $\boldsymbol{z} \in \mathbb{R}^\ell$. For a fixed $\gamma > 0$, the maximization problem (1) is then equivalent to the minimization of the regularized AE

$$\min_{\boldsymbol{\theta},\boldsymbol{\xi}} \frac{1}{n}\sum_{i=1}^n \Big\{\mathbb{E}_{\boldsymbol{\epsilon}}\|\boldsymbol{x}_i - \boldsymbol{\psi_\xi}(\boldsymbol{\phi_\theta}(\boldsymbol{x}_i) + \boldsymbol{\epsilon})\|_2^2 + \alpha\|\boldsymbol{\phi_\theta}(\boldsymbol{x}_i)\|_2^2\Big\}, \tag{2}$$

where $\alpha = \beta\sigma_0^2$, $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \gamma^2\mathbb{I})$ and where additive constants depending on γ have been omitted. The first term in (2) can be interpreted as an AE loss whereas the second term can be viewed as a regularization. This regularized AE interpretation motivates our method as introduced in Section 3.

The rest of the paper is organized as follows. In Section 2 we discuss the closely related work on disentangled representation learning and generation in the context of autoencoders. Further in Section 3 we describe the proposed model along with the connection between PCA and disentanglement. In Section 3.2 we discuss our contributions. In Section 4, we derive the evidence lower bound of the proposed model and show connections with the probabilistic models. In Section 5, we describe our experiments and discuss the results.

2 Related work

Related works can be broadly classified into two categories: Variational autoencoders (VAE) in the context of disentanglement and Restricted Kernel Machines (RKM), a recently proposed modeling framework that integrates kernel methods with deep learning.

VAE: As discussed in the introduction, (Higgins et al., 2017) suggested that a stronger emphasis on the posterior to match the factorized unit Gaussian prior puts further constraints on the implicit capacity of the latent bottleneck. C. P. Burgess et al. (2017) further analyzed the effect of the β term in depth. Later Chen et al. (2018) showed that the KL term includes the Mutual Information Gap which encourages disentanglement. Recently, several variants of VAEs promoting disentanglement have been proposed by adding extra terms to the ELBO. For instance, FactorVAE (Kim & Mnih, 2018) augments the ELBO by a new term enforcing factorization of the marginal posterior (or aggregate posterior). Rolínek et al. (2019) analyzed the reason for the alignment of the latent space with the coordinate axes, as the design of VAE itself does not suggest any such mechanism. The authors argue that due to the diagonal approximation in the encoder together with the inherent stochasticity forces the local orthogonality of the decoder. Locatello et al. (2020) considered adding an extra term that accounts for the knowledge of some partial label information to improve disentanglement. Later Ghosh et al. (2020) studied the deterministic AEs, where another quadratic regularization on the latent vectors was proposed. In contrast to Rolínek et al. (2019) where the implicit orthogonality of VAE was studied, our proposed model has orthogonality by design due to the introduction of the Stiefel manifold.

RKM: Restricted Kernel Machines (RKM) Suykens (2017) provides a represen-

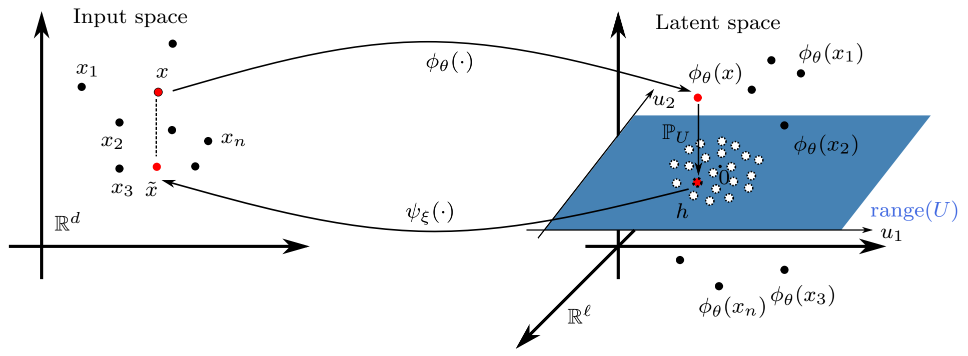


Figure 2: Schematic illustration of St-RKM training problem. The length of the dashed line represents the reconstruction error (see autoencoder term in (5)) and the length of the vector projecting on hyperplane represents the PCA reconstruction error. After training, the projected points tend to be distributed normally on the hyperplane.

tation of kernel methods with visible and hidden variables similar to the energy function of Restricted Boltzmann Machines (RBM) (LeCun et al., 2004; Hinton, 2005), thus linking kernel methods with RBMs. Training and prediction schemes are characterized by the stationary points for the unknowns in the objective. The equations in these stationary points lead to solving a linear-system or matrix decomposition for the training. Suykens (2017) shows various RKM formulations for doing classification, regression, kernel PCA and Singular Value Decomposition. Later the Kernel PCA formulation of RKM was extended to a multi-view generative model called Generative-RKM (Gen-RKM) which uses Convolutional Neural Networks as explicit feature maps Pandey et al. (2021, 2020). For the joint feature selection and subspace learning, the proposed training procedure performs eigendecomposition of the kernel/covariance matrix in every mini-batch of the optimization scheme. Intuitively, the model could be seen as learning an autoencoder with Kernel PCA in the bottleneck part. As a result, the computational complexity scales cubically with the mini-batch size and is proportional to the number of mini-batches. Moreover, backpropagation through the eigendecomposition could be numerically unstable due to the possibility of small eigenvalues. All such limitations are addressed by our proposed model.

3 Proposed mechanism

The main idea of this paper consists of learning an autoencoder along with finding an optimal linear subspace of the latent space such that the variance of the training set in latent space is maximized within this space. See Figure 2 to follow the discussion below. Note the distinction with linear autoencoders which also project the data into the low-dimensional subspace although via non-orthogonal transformations. As a consequence, the latent variables are not guaranteed to be uncorrelated. The encoder $\boldsymbol{\phi}_{\boldsymbol{\theta}} : \mathbb{R}^d \to \mathbb{R}^\ell$ typically sends input data to a latent space while the decoder $\boldsymbol{\psi}_{\boldsymbol{\xi}} : \mathbb{R}^\ell \to \mathbb{R}^d$ goes in the reverse direction, and constitutes an approximate inverse. Both the encoder and decoder are neural networks parameterized by vectors $\boldsymbol{\theta}$ and $\boldsymbol{\xi}$. However, it is unclear how to define a parametrization or an architecture of these neural networks so that the learned representation is disentangled. Therefore, in addition to these trained parameters, we also jointly find an m-dimensional linear subspace $\text{range}(U)$ of the latent space $\mathbb{R}^\ell$, such that the encoded training points mostly lie within this subspace. This linear subspace is given by the span of the orthonormal columns of the $\ell \times m$ matrix $U = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m]$. The set of such matrices with m orthonormal columns in $\mathbb{R}^\ell$ with $\ell \geq m$ defines the Stiefel manifold $\mathrm{St}(\ell, m)$. For a reference about optimization on Stiefel manifold, we refer to Absil et al. (2008). Input data is then encoded into a subspace of the latent space by

$$\boldsymbol{x} \mapsto \mathbb{P}_U \boldsymbol{\phi}_{\boldsymbol{\theta}}(x) = \boldsymbol{u}_1^\top \boldsymbol{\phi}_{\boldsymbol{\theta}}(x) \times \begin{bmatrix} | \\ \boldsymbol{u}_1 \\ | \end{bmatrix} + \cdots + \boldsymbol{u}_m^\top \boldsymbol{\phi}_{\boldsymbol{\theta}}(x) \times \begin{bmatrix} | \\ \boldsymbol{u}_m \\ | \end{bmatrix},$$

where the orthogonal projector onto $\mathrm{range}(U)$ is simply $\mathbb{P}_U = UU^\top$.

Orthogonal latent directions: Naturally, given an $m \times m$ orthogonal matrix O and a matrix $U \in \mathrm{St}(\ell, m)$, we have

$$\mathrm{range}(U) = \mathrm{range}(UO).$$

To select a specific matrix $U_\star = [\boldsymbol{u}_{\star,1}, \ldots, \boldsymbol{u}_{\star,m}] \in \mathrm{St}(\ell, m)$, we choose $\boldsymbol{u}_{\star,1}, \ldots, \boldsymbol{u}_{\star,m}$ to be the eigenvectors of the matrix $C_{\boldsymbol{\theta}} = \frac{1}{n}\sum_{i=1}^n \boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)\boldsymbol{\phi}_{\boldsymbol{\theta}}^\top(\boldsymbol{x}_i)$, associated with the m largest eigenvalues sorted in descending order. For simplicity, we assume that the m largest eigenvalues of $C_{\boldsymbol{\theta}}$ are distinct, whereas the general case involves minor technicalities. Here the feature map is assumed to be centered, i.e. $\mathbb{E}_{\boldsymbol{x} \sim p(\boldsymbol{x})}[\boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x})] = \boldsymbol{0}$, so that $C_{\boldsymbol{\theta}}$ is interpreted as a covariance matrix. Next, we state a result that will be used extensively later.

Proposition 1. *Let M be an $\ell \times \ell$ symmetric matrix. Let $\nu_1, \ldots, \nu_m$ be its m smallest eigenvalues, possibly including multiplicities, with associated orthonormal* *eigenvectors* $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m$*. Let* V *be a matrix whose columns are these eigenvectors. Then, the optimization problem* $\min_{U \in \mathrm{St}(\ell,m)} \mathrm{Tr}(U^\top M U)$ *has a minimizer at* $U_\star = V$ *and we have* ${U_\star}^\top M U_\star = \mathrm{diag}(\boldsymbol{\nu})$*, with* $\boldsymbol{\nu} = (\nu_1, \ldots, \nu_m)^\top$*.*

A few remarks are as follows. First, if $U_\star$ is a minimizer of the optimization problem in Proposition 1 then $U'_\star = U_\star O$ with O orthogonal is also a minimizer, but $U'^\top_\star M U'_\star$ is not necessarily diagonal. Second, notice that, if the eigenvalues of M in Proposition 1 have a multiplicity larger than 1, there can exist several sets of eigenvectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m$, associated to the m smallest eigenvalues, spanning distinct linear subspaces. Nevertheless, in practice, the eigenvalues of the matrices considered in this paper are numerically distinct.

Let us now use Proposition 1. For a given positive integer $m \leq \ell$, the subspace spanned by the eigenvectors of $C_{\boldsymbol{\theta}}$ with the m largest eigenvalues is obtained by solving

$$\min_{U \in \mathrm{St}(\ell,m)} \mathrm{Tr}\left(C_{\boldsymbol{\theta}} - \mathbb{P}_U C_{\boldsymbol{\theta}} \mathbb{P}_U\right) = \frac{1}{n} \sum_{i=1}^{n} \|\mathbb{P}_{U^\perp} \boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)\|_2^2,$$

where $\mathbb{P}_{U^\perp} = \mathbb{I} - \mathbb{P}_U$, as it is explained, for instance, in Section 4.1 of Avron et al. (2014). The above objective corresponds to the reconstruction error of Kernel PCA, for the kernel $k_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{\phi}_{\boldsymbol{\theta}}^\top(\boldsymbol{x}) \boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{y})$. As described earlier, we choose a specific $U_\star \in \mathrm{St}(\ell, m)$ by requiring that the following matrix is diagonal

$$U_\star^\top C_{\boldsymbol{\theta}} U_\star = \mathrm{diag}(\boldsymbol{\lambda}), \tag{3}$$

where $\boldsymbol{\lambda}$ is a vector containing the m largest eigenvalues sorted in decreasing order. If these eigenvalues are distinct, then the $U_\star$ is essentially unique, up to sign flip of each of its columns. Notice that $\mathrm{Tr}(U_\star^\top C_{\boldsymbol{\theta}} U_\star) = \mathrm{Tr}(U_\star U_\star^\top C_{\boldsymbol{\theta}} U_\star U_\star^\top)$.

Orthogonal directions of variation in input space: We want the lines defined by the orthonormal vectors $\{\boldsymbol{u}_{\star,1}, \dots, \boldsymbol{u}_{\star,m}\}$ to provide directions associated with different generative factors of our model. In other words, we conjecture that a possible formalization of disentanglement is that *the principal directions in latent space match orthogonal directions of variation in the data space* (see Figure 2). That is to say, we would like that

$$U_\star^\top \sum_{a=1}^{d} \left(\nabla \boldsymbol{\psi}_a(\boldsymbol{y}_i) \nabla \boldsymbol{\psi}_a(\boldsymbol{y}_i)^\top \right) U_\star \text{ is diagonal,} \tag{4}$$

for all the points in latent space $\boldsymbol{y}_i = \mathbb{P}_U \boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$ for $i = 1, \dots, n$. In (4), $\boldsymbol{\psi}_a(\boldsymbol{y})$ refers to the a-th component of the image $\boldsymbol{\psi}(\boldsymbol{y}) \in \mathbb{R}^d$. To sketch this idea, we study below the local motions in the latent space.

Let $\Delta_k = \nabla \boldsymbol{\psi}(\boldsymbol{y})^\top \boldsymbol{u}_{\star,k} \in \mathbb{R}^d$ be the directional derivative of $\boldsymbol{\psi}$ at point $\boldsymbol{y}$ in the direction $\boldsymbol{u}_{\star,k}$ with $1 \leq k \leq m$. Then, as one moves in the latent space from a point $\boldsymbol{y}$ in the direction of $\boldsymbol{u}_{\star,k}$, the generated data changes by

$$\boldsymbol{\psi}(\boldsymbol{y} + t\boldsymbol{u}_{\star,k}) - \boldsymbol{\psi}(\boldsymbol{y}) = t\Delta_k + \mathcal{O}(t^2),$$

with $\Delta_k \in \mathbb{R}^d$ and $t \in \mathbb{R}$. Consider now a different direction, i.e., $k' \neq k$. As the latent point moves along $\boldsymbol{u}_{\star,k}$ or along $\boldsymbol{u}_{\star,k'}$, we expect the decoder output to vary in a significantly different manner, i.e., $\Delta_k^\top \Delta_{k'} = 0$. We presume this interpretation to model the change in floor color and object scale in Figure 1 for instance. More explicitly, we can expect $\boldsymbol{u}_k$ and $\boldsymbol{u}_{k'}$ to model, respectively, the change of colors of the floor and of the main object while leaving the color of the other objects unchanged. Since the floor and the main object do not overlap, that is, are different regions in pixel space, we would have $\Delta_k^\top \Delta_{k'} = 0$. Admittedly, the change in object shape in Figure 1 is less obviously interpreted. Now, denote by Δ the matrix obtained by stacking the vector Δ_k as columns for $1 \leq k \leq m$. Explicitly, we have $\Delta = \nabla \boldsymbol{\psi}_a(\boldsymbol{y})^\top U_\star$. *Hence, for all $\boldsymbol{y}$ in the latent space, we expect the Gram matrix $\Delta^\top \Delta$ to be diagonal* (cf. (4)). We now discuss how this idea might be realized by minimizing specific objective functions.

3.1 Objective function

In this paper, we propose to train an objective function which is composed of (i) an AE loss and (ii) a PCA loss. Hence, the proposed model is given by

$$\min_{\substack{U \in \mathrm{St}(\ell,m)\\ \boldsymbol{\theta},\boldsymbol{\xi}}} \underbrace{\lambda \frac{1}{n} \sum_{i=1}^{n} L_{\boldsymbol{\xi},\mathbb{P}_U}\left(\boldsymbol{x}_i, \boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)\right)}_{\text{Autoencoder objective}} + \underbrace{\mathrm{Tr}\left(C_{\boldsymbol{\theta}} - \mathbb{P}_U C_{\boldsymbol{\theta}} \mathbb{P}_U\right)}_{\text{PCA objective}}, \tag{5}$$

where $\lambda > 0$ is a trade-off parameter and $C_{\boldsymbol{\theta}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)\boldsymbol{\phi}_{\boldsymbol{\theta}}^\top(\boldsymbol{x}_i)$. Naturally, the above objective is invariant if U is replaced by UO with O an orthogonal matrix. Given a local minimizer, we select $U_\star \in \mathrm{St}(\ell, m)$ such that $U_\star^\top C_{\boldsymbol{\theta}} U_\star$ is diagonal as in equation (3) above, to identify the principal directions in the latent space. This last step is conveniently done with a singular value decomposition; see step 10 of Algorithm 1. In the proposed model, reconstruction of an out-of-sample point $\boldsymbol{x}$ is given by $\boldsymbol{\psi}_{\boldsymbol{\xi}}\big(\mathbb{P}_U \boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x})\big)$. We call the procedure to

$$\text{find a triplet } (U_\star, \boldsymbol{\theta}, \boldsymbol{\xi}) \text{ solving (5) s.t. } U_\star^\top C_{\boldsymbol{\theta}} U_\star \text{ is diagonal,} \tag{St-RKM}$$

the training of a Stiefel-Restricted Kernel Machines (5) in view of our discussion in Section 2. The basic idea is to design different AE losses with a regularization term that penalizes the feature map in the orthogonal subspace $U^\perp$. The choice of the AE losses is motivated by the expression of the regularized AE in (2) and by the following Lemma which extends the result of Rolínek et al. (2019). Here we adapt it in the context of optimization on the Stiefel manifold; see Appendix for the proof.

Lemma 1. *Let $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbb{I}_m)$ a random vector and $U \in \mathrm{St}(\ell, m)$. Let $\boldsymbol{\psi}_a(\cdot) \in \mathcal{C}^2(\mathbb{R}^\ell)$ with $a \in [d]$. If the function $[\boldsymbol{\psi}(\cdot) - \boldsymbol{x}]_a^2$ has L_a-Lipschitz continuous Hessian for all $a \in [d]$, we have*

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\epsilon}} \|\boldsymbol{x} - \boldsymbol{\psi}(\boldsymbol{y} + \sigma U \boldsymbol{\epsilon})\|_2^2 = \|\boldsymbol{x} - \boldsymbol{\psi}(\boldsymbol{y})\|_2^2 &+ \sigma^2 \operatorname{Tr}\left(U^\top \nabla \boldsymbol{\psi}(\boldsymbol{y}) \nabla \boldsymbol{\psi}(\boldsymbol{y})^\top U\right) \\ &- \sigma^2 \sum_{a=1}^{d} [\boldsymbol{x} - \boldsymbol{\psi}(\boldsymbol{y})]_a \operatorname{Tr}\left(U^\top \operatorname{Hess}_{\boldsymbol{y}}[\boldsymbol{\psi}_a] U\right) + \sum_{a=1}^{d} R_a(\sigma),\end{aligned} \tag{6}$$

with $|R_a(\sigma)| \leq \frac{1}{6}\sigma^3 L_a \frac{\sqrt{2}(m+1)\Gamma((m+1)/2)}{\Gamma(m/2)}$ where Γ is Euler's Gamma function.

A few remarks are as follows. In Lemma 1, the first term on the right-hand side in (6) plays the role of the classical AE loss. The second term is proportional to the trace of (4). This is related to our discussion above where we argue that jointly diagonalizing both $U^\top \nabla \boldsymbol{\psi}(\boldsymbol{y}) \nabla \boldsymbol{\psi}(\boldsymbol{y})^\top U$ and $U^\top C_{\boldsymbol{\theta}} U$ helps to enforce disentanglement. However, determining the behavior of the third term in (6) is difficult. This is because, for a typical neural network architecture, it is unclear in practice if the function $[\boldsymbol{x} - \boldsymbol{\psi}(\cdot)]_a^2$ has L_a-Lipschitz continuous Hessian for all $a \in [d]$. Hence we propose below another AE loss (splitted loss) in order to cancel the third term in (6). Nevertheless, the assumption in Lemma 1 is used to provide a meaningful bound on the remainder in (6). In the light of these remarks, we propose two stochastic AE losses as described below.

AE losses: In analogy with the VAE objective (2), the first AE encoder loss function can be chosen as

$$L^{(\sigma)}_{\boldsymbol{\xi},\mathbb{P}_U}(\boldsymbol{x},\boldsymbol{z}) = \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbb{I}_m)} \left\|\boldsymbol{x} - \boldsymbol{\psi}_{\boldsymbol{\xi}}\big(\mathbb{P}_U\boldsymbol{z} + \sigma U\boldsymbol{\epsilon}\big)\right\|_2^2, \text{ with } \sigma > 0.$$

As motivated by Lemma 1 above, the noise term $\sigma U\boldsymbol{\epsilon}$ above promotes a *smoother* decoder network. To further promote disentanglement, we propose a splitted AE loss

$$L^{(\sigma),sl}_{\boldsymbol{\xi},\mathbb{P}_U}(\boldsymbol{x},\boldsymbol{z}) = \left\|\boldsymbol{x} - \boldsymbol{\psi}_{\boldsymbol{\xi}}\big(\mathbb{P}_U\boldsymbol{z}\big)\right\|_2^2 + \mathbb{E}_{\boldsymbol{\epsilon}}\left\|\boldsymbol{\psi}_{\boldsymbol{\xi}}\big(\mathbb{P}_U\boldsymbol{z}\big) - \boldsymbol{\psi}_{\boldsymbol{\xi}}\big(\mathbb{P}_U\boldsymbol{z} + \sigma U\boldsymbol{\epsilon}\big)\right\|_2^2, \qquad (7)$$

with $\boldsymbol{\epsilon} \sim \mathcal{N}(0,\mathbb{I}_m)$. The first term in (7) is the classical AE loss while the second term promotes orthogonal directions of variations. Thus, by relating Lemma 1 to (7), we see that

$$L^{(\sigma),sl}_{\boldsymbol{\xi},\mathbb{P}_U}(\boldsymbol{x},\boldsymbol{z}) = \left\|\boldsymbol{x} - \boldsymbol{\psi}_{\boldsymbol{\xi}}\big(\mathbb{P}_U\boldsymbol{z}\big)\right\|_2^2 + \sigma^2 \operatorname{Tr}\left(U^\top \nabla\boldsymbol{\psi}(\boldsymbol{y})\nabla\boldsymbol{\psi}(\boldsymbol{y})^\top U\right) + \sum_{a=1}^{d} R_a(\sigma).$$

In short, the optimization over U in (5) with the splitted loss aims to promote a $U_\star$ such that

$$U_\star^\top C_{\boldsymbol{\theta}} U_\star \text{ and } U_\star^\top \left(\textstyle\sum_{i=1}^n \nabla\boldsymbol{\psi}(\boldsymbol{y}_i)\nabla\boldsymbol{\psi}(\boldsymbol{y}_i)^\top\right) U_\star \text{ are } \textit{jointly diagonal.}$$

Figure 3 gives a visualization of the diagonal form of

$$\frac{1}{|\mathcal{C}|}\sum_{i\in\mathcal{C}} U_\star^\top \nabla\boldsymbol{\psi}(\boldsymbol{y}_i)\nabla\boldsymbol{\psi}(\boldsymbol{y}_i)^\top U_\star, \text{ with } \boldsymbol{y}_i = \mathbb{P}_U\boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}_i) \qquad (8)$$

obtained after training; where $\mathcal{C}$ contains the indices of a subset of 50 images sampled uniformly at random. For numerical values, Table 6 in the appendix shows the normalized diagonalization errors.

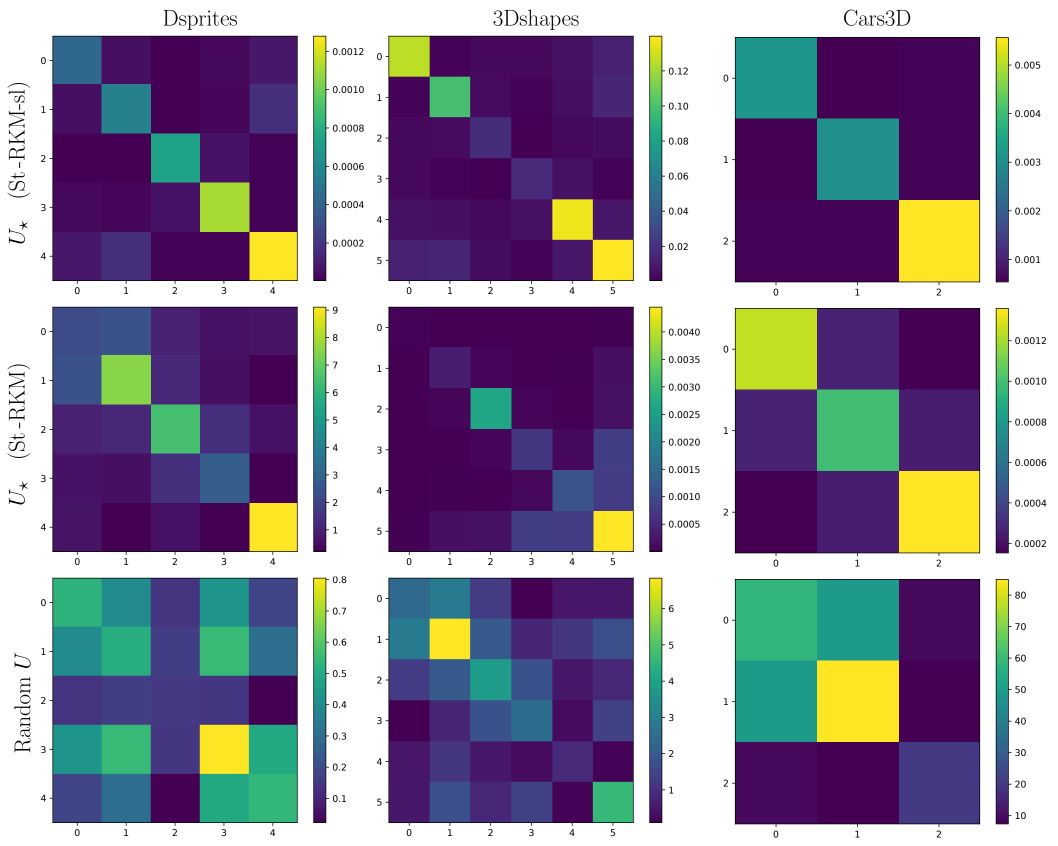


Figure 3: Visualizing the matrix (8) for St-RKM models after training on three datasets. The first two rows show (8) where $U = U_\star \in \mathrm{St}(\ell, m)$ is the output of Algorithm 1. These matrices are effectively close to being diagonal and especially for St-RKM-sl, as it is expected. In contrast, the third row shows the same matrix (8) with $U \in \mathrm{St}(\ell, m)$ sampled uniformly at random; see Table 6 for the corresponding normalized diagonalization errors.

Note that we do not simply propose another encoder-decoder architecture, given by: $U^\top \boldsymbol{\phi}_{\boldsymbol{\theta}}(\cdot)$ and $\boldsymbol{\psi}_{\boldsymbol{\xi}}(U\cdot)$. Instead, our objective assumes that the neural network defining the encoder provides a better embedding if we impose that it maps training points on a linear subspace of dimension $m < \ell$ in the ℓ-dimensional latent space. In other words, the optimization of the parameters in the last layer of the encoder does not play a redundant role, since the second term in (5) clearly also depends on $\mathbb{P}_{U^\perp}\boldsymbol{\phi}_{\boldsymbol{\theta}}(\cdot)$. The full training involves an alternating minimization procedure which is described in Algorithm 1.

3.2 Contributions

Here is a summary of our contributions. We propose two main changes with respect to the related works: (*i*) To promote disentangled representation learning, we propose orthogonal projection in the latent space via a rectangular matrix which is valued on the Stiefel manifold. Then for the training, we use the Cayley ADAM algorithm of Li et al. (2020) for stochastic optimization on the Stiefel manifold and call our proposed model St-RKM. (*ii*) We propose several objective functions to learn the feature map and the pre-image map networks in the form of an encoder and a decoder respectively. The best configuration for promoting a disentangled representation is

$$\min_{\substack{U\in\mathrm{St}(\ell,m)\\ \boldsymbol{\theta},\boldsymbol{\xi}}} \frac{\lambda}{n}\sum_{i=1}^{n}\text{(splitted) AE loss}(\boldsymbol{x}_i,\mathbb{P}_U,\boldsymbol{\theta},\boldsymbol{\xi}) + \text{PCA objective}(C_{\boldsymbol{\theta}},\mathbb{P}_U),$$

where the covariance matrix reads $C_{\boldsymbol{\theta}} = \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)\boldsymbol{\phi}_{\boldsymbol{\theta}}^\top(\boldsymbol{x}_i)$ and $\mathbb{P}_U = UU^\top$ with U an $\ell\times m$ matrix with orthonormal columns. Here $\lambda > 0$ is a trade-off parameter. The final parameters $(U_\star,\boldsymbol{\theta},\boldsymbol{\xi})$ give a local minimizer of this objective with $U_\star$ chosen such that $U_\star^\top C_{\boldsymbol{\theta}} U_\star$ is diagonal. (*iii*) We validate through experiments the following statement: the combination of a splitted AE loss with a PCA objective by using an explicit optimization on the Stiefel manifold promotes disentanglement. In this paper, disentanglement is interpreted as jointly diagonalizing the matrix representing variations in the input space with respect to latent motions

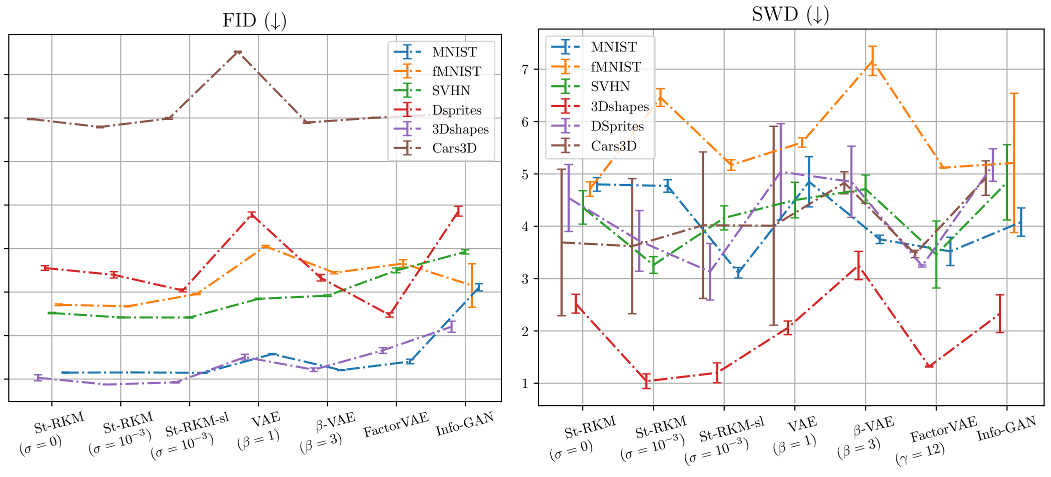


Figure 4: Fréchet Inception Distance (FID) Heusel et al. (2017) and Sliced Wasserstein Distance (SWD) scores (mean and 1 standard-deviation) for 8000 randomly generated samples (smaller is better).

$\sum_i U_\star^\top \nabla \boldsymbol{\psi_\xi}(\boldsymbol{y}_i) \nabla \boldsymbol{\psi_\xi}(\boldsymbol{y}_i)^\top U_\star$ where $\boldsymbol{y}_i = \mathbb{P}_{U_\star} \boldsymbol{\phi_\theta}(\boldsymbol{x}_i)$ and the covariance matrix of the dataset in the latent space $U_\star^\top C_{\boldsymbol{\theta}} U_\star$.

4 Connections with the Evidence Lower Bound

We now discuss the interpretation of the proposed model in the probabilistic setting and the independence of latent factors. In order to formulate an ELBO, consider the following random encoders:

$$q(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}|\boldsymbol{\phi_\theta}(\boldsymbol{x}), \gamma^2 \mathbb{I}_\ell) \text{ and } q_U(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}|\mathbb{P}_U \boldsymbol{\phi_\theta}(\boldsymbol{x}), \sigma^2 \mathbb{P}_U + \delta^2 \mathbb{P}_{U^\perp}),$$

where $\boldsymbol{\phi_\theta}$ has zero mean on the data distribution. Here, σ^2 plays the role of a trade-off parameter, while the regularization parameter δ is introduced for technical reasons and is put to a numerically small absolute value (see Appendix for details). Let the decoder be $p(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\psi_\xi}(\boldsymbol{z}), \sigma_0^2 \mathbb{I})$ and the latent space distribution is parametrized by $p(\boldsymbol{z}) = \mathcal{N}(0, \Sigma)$ where $\Sigma \in \mathbb{R}^{\ell\times\ell}$ is a covariance matrix. We treat Σ as a parameter of the optimization problem that is determined at the last stage of the training. Then the minimization problem (5) with stochastic AE loss is equivalent to the maximization of

$$\frac{1}{n}\sum_{i=1}^{n}\Big\{\underbrace{\mathbb{E}_{q_U(\boldsymbol{z}|\boldsymbol{x}_i)}[\log(p(\boldsymbol{x}_i|\boldsymbol{z}))]}_{\text{(I)}} - \underbrace{\mathrm{KL}(q_U(\boldsymbol{z}|\boldsymbol{x}_i), q(\boldsymbol{z}|\boldsymbol{x}_i))}_{\text{(II)}} - \underbrace{\mathrm{KL}(q_U(\boldsymbol{z}|\boldsymbol{x}_i), p(\boldsymbol{z}))}_{\text{(III)}}\Big\}, \tag{9}$$

which is a lower bound to the ELBO, since the KL divergence in (II) is positive. For details of the derivation, see Appendix. The hyper-parameters γ, σ, σ_0 take a fixed value. Up to additive constants, the terms (I) and (II) of (9) match the objective (5). The third term (III) in (9) is optimized after the training of the first two terms. It can be written as follows

$$\frac{1}{n}\sum_{i=1}^{n}\mathrm{KL}(q_U(\boldsymbol{z}|\boldsymbol{x}_i), p(\boldsymbol{z})) = \frac{1}{2}\operatorname{Tr}[\Sigma_0\Sigma^{-1}] + \frac{1}{2}\log(\det\Sigma) + \text{constants},$$

with $\Sigma_0 = \mathbb{P}_U C_{\boldsymbol{\theta}}\mathbb{P}_U + \sigma^2\mathbb{P}_U + \delta^2\mathbb{P}_{U^\perp}$. Hence, in that case, the optimal covariance matrix is diagonalized $\Sigma = U(\operatorname{diag}(\boldsymbol{\lambda}) + \sigma^2\mathbb{I}_m)U^\top + \delta^2\mathbb{P}_{U_\perp}$, with $\boldsymbol{\lambda}$ denoting the principal values of the PCA.

Now we briefly discuss the factorization of the encoder. Let $\boldsymbol{h}(\boldsymbol{x}) = U^\top\boldsymbol{\phi_\theta}(\boldsymbol{x})$ and let the 'effective' latent variable be $\boldsymbol{z}^{(U)} = U^\top\boldsymbol{z} \in \mathbb{R}^m$. Then the probability density function of $q_U(\boldsymbol{z}|\boldsymbol{x})$ is

$$f_{q_U(\boldsymbol{z}|\boldsymbol{x})}(\boldsymbol{z}) = \frac{e^{-\frac{\|U_\perp^\top\boldsymbol{z}\|_2^2}{2\delta^2}}}{(\sqrt{2\pi\delta^2})^{\ell-m}}\prod_{j=1}^{m}\frac{e^{-\frac{(\boldsymbol{z}_j^{(U)}-\boldsymbol{h}_j(\boldsymbol{x}))^2}{2\sigma^2}}}{\sqrt{2\pi\sigma^2}},$$

where the first factor is approximated by a Dirac delta if $\delta \to 0$. Hence, the factorized form of q_U shows the independence of the latent variables $\boldsymbol{z}^{(U)}$. This factorization is used as a regularization term in the objective by Kim & Mnih (2018) to promote disentanglement. In particular, the term (II) in (9) is analogous to a 'Total Correlation' loss (Chen et al., 2018).

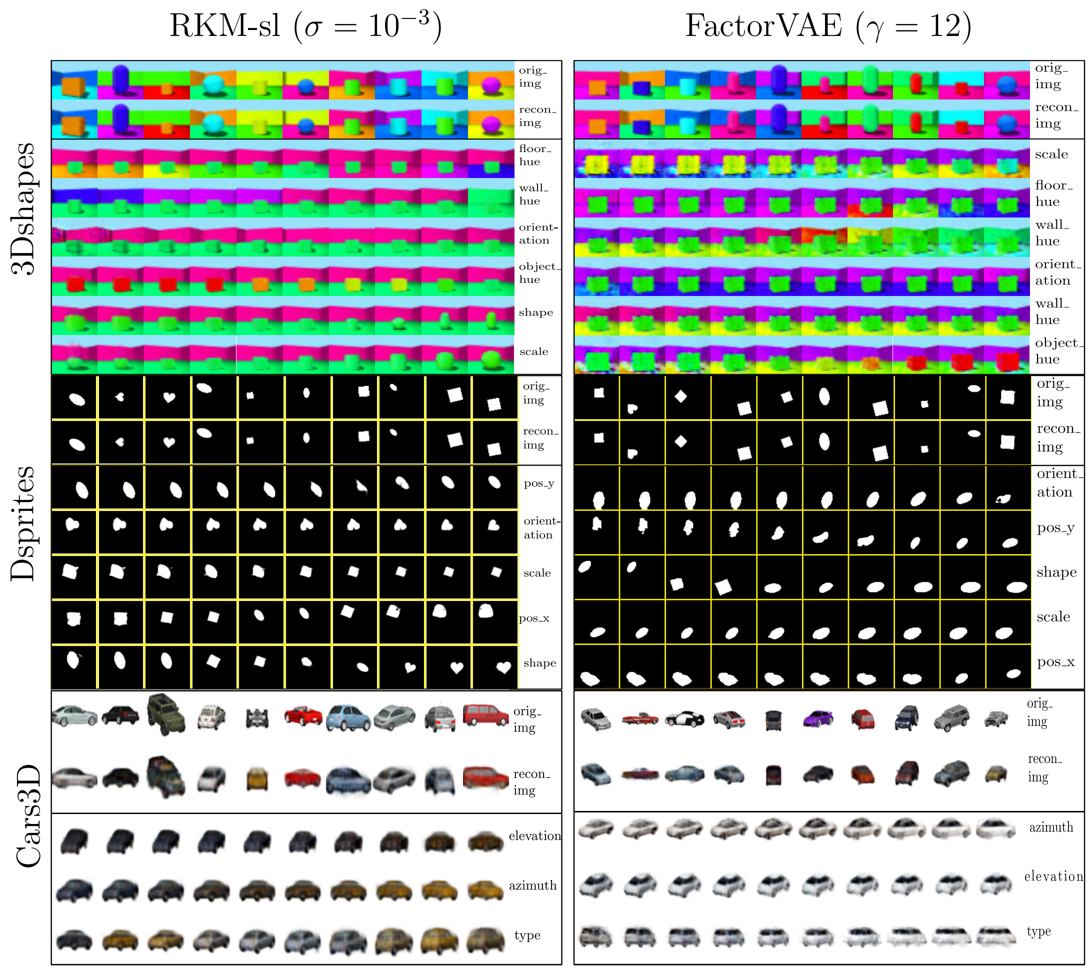


Figure 5: Traversals along the principal components. First-two rows show the ground-truth and reconstructed images. Further, each subsequent row shows the generated images by traversing along a principal component in the latent space. The last column in each sub-image indicates the dominant factor of variation.

5 Experiments

In this section, we investigate if St-RKM[2] can simultaneously achieve (i) accurate reconstructions on training data (ii) good random generations, and (iii) good disentanglement performance. We use the standard datasets: MNIST LeCun & Cortes (2010), Fashion-MNIST (Xiao et al., 2017) (fMNIST), and SVHN (Netzer et al., 2011). To evaluate disentanglement, we use datasets with known ground-truth generating factors such as Dsprites (Matthey et al., 2017), 3Dshapes (C. Burgess & Kim, 2018), and Cars3D (Reed et al., 2015). Further, all figures and tables report average errors with 1 standard deviation over 10 experiments.

Table 1: Training time in minutes (for 1000 epochs, mean with 1 standard deviation over 10 runs) and the number of parameters (Nb) of the generative models on the MNIST dataset.

Model	St-**RKM**	(β)-**VAE**	**FactorVAE**	**Info-GAN**
Nb parameters	**4164519**	4165589	8182591	4713478
Training time	21.93 (1.3)	**19.83** (0.8)	33.31 (2.7)	45.96 (1.6)

Algorithm: We use an alternating-minimization scheme as shown in Algorithm 1. First, the Adam optimizer with a learning rate 2×10^{-4} is used to update the encoder-decoder parameters and then, the Cayley Adam optimizer (Li et al., 2020) with a learning rate 10^{-4} is used to update U. Finally, at the end of the training, we recompute U from the Singular Value Decomposition (SVD) of the covariance matrix as a final correction-step of the Kernel PCA term in our objective (step 10 of

[2]The source code is available at http://bit.ly/StRKM_code

Algorithm 1). Since the $\ell \times \ell$ covariance matrix is typically small, this decomposition is fast (see Table 3). In practice, our training procedure only marginally increases the computation cost which can be seen from training times in Table 1.

Algorithm 1 Manifold optimization of St-RKM

```
Input: $\{\boldsymbol{x}_i\}_{i=1}^n$, $\boldsymbol{\phi_\theta}, \boldsymbol{\psi_\zeta}, \mathcal{J} :=$ Eq. 5
Output: Learned $\boldsymbol{\theta}, \boldsymbol{\zeta}, U$
1: procedure TRAIN
2:     while not converged do
3:         $\{\boldsymbol{x}\} \leftarrow \{$Get mini-batch$\}$
4:         Get embeddings $\boldsymbol{\phi_\theta}(\boldsymbol{x}) \leftarrow \boldsymbol{x}$
5:         Compute centered $C_{\boldsymbol{\theta}}$                    ▷ Covariance matrix
6:         Update $\{\boldsymbol{\theta_e}, \boldsymbol{\psi_g}\} \leftarrow$ Adam($\mathcal{J}$)        ▷ Optimization step
7:         Update $\{U\} \leftarrow$ Cayley_Adam($\mathcal{J}$)          ▷ Optimization step
8:     end while
9:     Do steps 4-5 over whole dataset
10:    $U \leftarrow$ SVD($C_{\boldsymbol{\theta}}$)                                ▷ Equation (3)
11: end procedure
```

Experimental setup: We consider four baselines for comparison: (i) VAE, (ii) β-VAE, (iii) FactorVAE and (iv) Info-GAN. An ablation study with the Gen-RKM is shown in the appendix D. Extensive experimentation was not computationally feasible since the evaluation and decomposition of kernel matrices scales $\mathcal{O}(n^2)$ and $\mathcal{O}(n^3)$ with the dataset size; see the discussion in Section 2.

Inductive biases: To be consistent in evaluation, we keep the same encoder (discriminator) and decoder (generator) architecture; and the same latent dimension across the models. We use convolutional neural networks due to the choice of image datasets for evaluating generation and disentanglement. In the case of Info-GAN, batch-normalization is added for training stability; see appendix C for details. For the determination of the hyperparameters of other models, we start from values in the range of the parameters suggested in the authors' reference implementation. After trying various values we noticed that $\beta = 3$ and $\gamma = 12$ seem to work well across the datasets that we considered for β-VAE and FactorVAE respectively. Furthermore, in all the experiments on St-RKM, we keep the reconstruction weight $\lambda = 1$. All models are trained on the entire dataset. Note that for the same encoder-decoder network, the St-RKM model has the least number of parameters compared to any VAE variants and Info-GAN (see Table 1).

To evaluate the quality of generated samples, we report the Fréchet Inception Distance (Heusel et al., 2017) (FID) and the Sliced Wasserstein Distance (SWD) (Karras et al., 2017) scores with mean and standard deviation in Figure 4. Note that FID scores are not necessarily appropriate for Dsprites since this dataset is significantly different from ImageNet on which the Inception network was originally trained. Randomly generated samples are shown in Figure 8 in Appendix. To generate samples from the deterministic St-RKM ($\sigma = 0$), we sample from a fitted normal distribution on the latent embedding of the dataset; for a similar prodedure, see Ghosh et al. (2020)). Figure 4 shows that the St-RKM variants perform better (lower mean scores) on most datasets and within them, the stochastic variants with $\sigma = 10^{-3}$ perform best. This can be attributed to

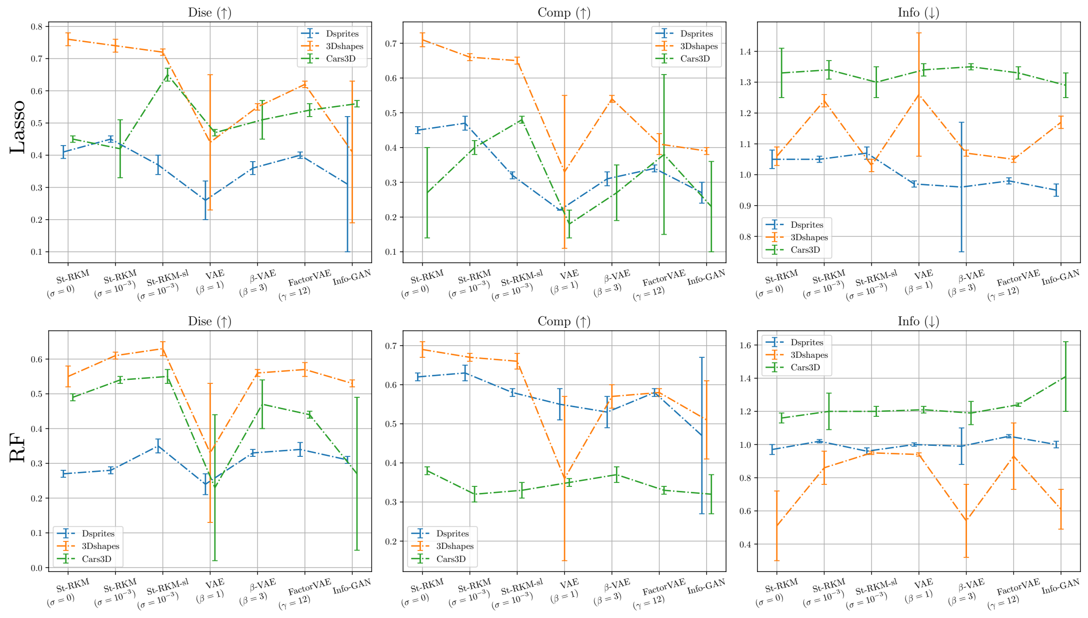


Figure 6: Eastwood framework's Eastwood & Williams (2018) disentanglement metric with Lasso and Random Forest (RF) regressor. The plot shows mean and 1 standard-deviation of scores over 10 iterations. For disentanglement and completeness higher score is better, for informativeness, lower is better. 'Info. indicates (average) root-mean-square error in predicting $\boldsymbol{z}$.

a better generalization of the decoder network due to the addition of noise-term on latent-variables; see Lemma 1. The training times for St-RKM variants are shorter compared to FactorVAE and Info-GAN due to a significantly small number of parameters.

To evaluate the disentanglement performance, various metrics have been proposed. A comprehensive review by Locatello et al. (2019) shows that the various disentanglement metrics are correlated albeit with a different degree of correlation across datasets. In this paper, we use three metrics to evaluate disentanglement, namely, Eastwood's framework (Eastwood & Williams, 2018), Mutual Information Gap (MIG) (Chen et al., 2018) and Separated Attribute Predictability (SAP) (Ku-

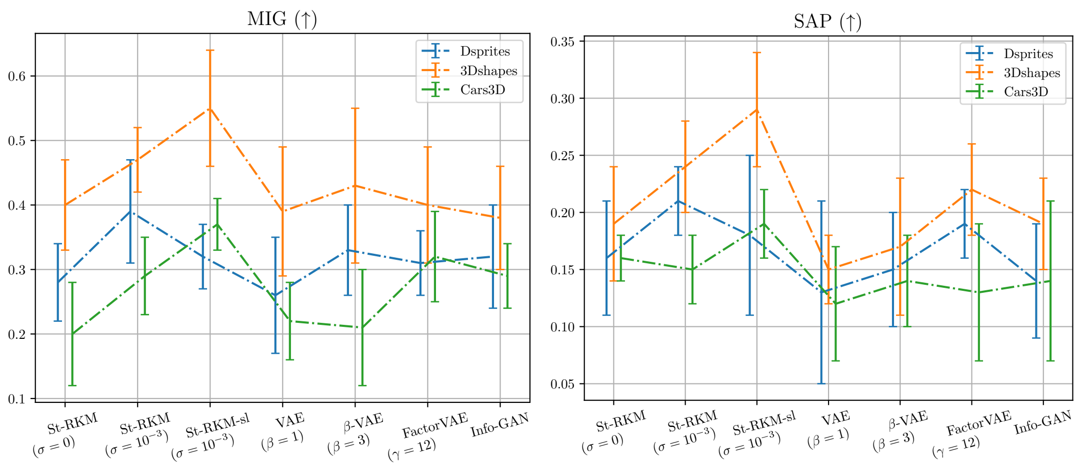


Figure 7: MIG (Chen et al., 2018; Locatello et al., 2019) and SAP (Kumar et al., 2018) scores to evaluate disentanglement performance showing the mean (standard deviation) over 10 random seeds.

mar et al., 2018) scores. Eastwood's framework (Eastwood & Williams, 2018) further proposes three metrics: *disentanglement*: the degree to which a representation factorizes the underlying factors of variation, with each variable capturing at most one generative factor; *completeness*: the degree to which each underlying factor is captured by a single code variable; and *informativeness*: the amount of information that a representation captures about the underlying factors of variation. Further we use a slightly modified version of MIG score as proposed by Locatello et al. (2019). Figure 6 shows that St-RKM variants have better disentanglement and completeness scores (higher mean scores). However, the informativeness scores are higher for St-RKM when using a lasso-regressor in contrast to mixed scores with a Random forest regressor. Figure 7 further complements these observations by showing MIG and SAP scores. Here the St-RKM-sl model has the highest mean scores for every dataset. Qualitative assessment can be done from Figure 5 which shows the generated images by traversing along the principal components in the latent space. In the 3Dshapes dataset, the St-RKM model captures floor-hue, wall-hue, and orientation perfectly but has a slight entanglement in capturing other factors. This is worse in β-VAE which has entanglement in all dimensions except the floor-hue along with noise in some generated images. Similar trends can be observed in the Dsprites and Cars3D datasets.

Conclusion

This paper proposes St-RKM model for disentangled representation learning and generation based on manifold optimization. For the training, we use the Cayley Adam algorithm of Li et al. (2020) for stochastic optimization on the Stiefel manifold. Computationally, St-RKM only increases the training time by a reasonably small amount compared to β-VAE for instance. Further, we propose several autoencoder objectives and discuss that the combination of a stochastic AE loss with an explicit optimization on the Stiefel manifold promotes disentanglement. Additionally, we establish connections with probabilistic models, formulate an Evidence Lower Bound, and discuss the independence of latent factors. Where the considered baselines have a trade-off between generation quality and disentanglement, we improve on both these aspects as illustrated through various experiments. Some limitations of the proposed model are as follows. A first limitation is hyperparameter selection: the number of components in the KPCA, neural network architecture and the final size of the feature map. When additional knowledge on the data is available, we suggest that the user selects the number of components close to the number of underlying generating factors. The final size of the feature map should be large enough so that KPCA extracts meaningful components. Second, we interpret the disentanglement as the two orthogonal changes in the latent space correspond to two orthogonal changes in input space. Although not perfect, we believe this is a reasonable mathematical approximation of the loosely defined notion of ‘disentanglement’. Moreover, experimental results confirm this assumption. Among the possible regularizers on the hidden features, the model associated to the squared Euclidean norm was analyzed in detail, while a deeper study of other regularizers is a prospect for further research, in particular for the case of spherical units.

Acknowledgments

Most of this work was done when Michaël Fanuel was at KU Leuven. EU: The research leading to these results has received funding from the European Research Council under the European Union’s Horizon 2020 research and innovation program / ERC Advanced Grant E-DUALITY (787960). This paper reflects only the authors’ views and the Union is not liable for any use that may be made of the contained information. Research Council KUL: Optimization frameworks for deep kernel machines C14/18/068 Flemish Government: FWO: projects: GOA4917N (Deep Restricted Kernel Machines: Methods and Foundations), PhD/Postdoc grant Impulsfonds AI: VR 2019 2203 DOC.0318/1QUATER Kenniscentrum Data en Maatschappij This research received funding from the Flemish Government (AI Research Program). The authors are affiliated to Leuven.AI - KU Leuven institute for AI, B-3000, Leuven, Belgium. Ford KU Leuven Research Alliance Project KUL0076 (Stability analysis and performance improvement of deep reinforcement learning algorithms)s), ICT 48 TAILOR, Leuven.AI Institute. The computational resources and services used in this work were provided by the VSC (Flemish Supercomputer Center), funded by the Research Foundation - Flanders (FWO) and the Flemish Government department EWI.

Appendix

A Proof of Lemma 1

We first quote a result that is used in the context of optimization ((Nesterov, 2014), Lemma 1.2.4). Let f a function with L_a-Lipschitz continuous Hessian. Then,

$$\Big|\underbrace{f(\boldsymbol{y}_1) - f(\boldsymbol{y}) - \nabla f(\boldsymbol{y})^\top(\boldsymbol{y}_1 - \boldsymbol{y}) - \frac{1}{2}(\boldsymbol{y}_1 - \boldsymbol{y})^\top \operatorname{Hess}_{\boldsymbol{y}}[f](\boldsymbol{y}_1 - \boldsymbol{y})}_{r(\boldsymbol{y}_1 - \boldsymbol{y})}\Big| \leq \frac{L_a}{6}\|\boldsymbol{y}_1 - \boldsymbol{y}\|_2^3. \tag{10}$$

Then, we calculate the power series expansion of $f(\boldsymbol{y}) = [\boldsymbol{x} - \boldsymbol{\psi}(\boldsymbol{y})]_a^2$ and take the expectation with respect to $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbb{I})$. First, we have $\nabla f(\boldsymbol{y}) = -2[\boldsymbol{x} - \boldsymbol{\psi}(\boldsymbol{y})]_a \nabla \boldsymbol{\psi}_a(\boldsymbol{y})$ and

$$\operatorname{Hess}_{\boldsymbol{y}}[f] = 2\nabla\boldsymbol{\psi}_a(\boldsymbol{y})\nabla\boldsymbol{\psi}_a(\boldsymbol{y})^\top - 2[\boldsymbol{x} - \boldsymbol{\psi}(\boldsymbol{y})]_a \operatorname{Hess}_{\boldsymbol{y}}[\boldsymbol{\psi}_a].$$

Then, we use (10) with $\boldsymbol{y}_1 - \boldsymbol{y} = \sigma U\boldsymbol{\epsilon}$. By taking the expectation over $\boldsymbol{\epsilon}$, notice that the order 1 term in σ vanishes since $\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{\epsilon}] = 0$. We find

$$\begin{aligned}\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{x} - \boldsymbol{\psi}(\boldsymbol{y} + \sigma U\boldsymbol{\epsilon})]_a^2 =& [\boldsymbol{x} - \boldsymbol{\psi}(\boldsymbol{y})]_a^2 + \sigma^2 \operatorname{Tr}\left(U^\top \nabla\boldsymbol{\psi}_a(\boldsymbol{y})\nabla\boldsymbol{\psi}_a(\boldsymbol{y})^\top U\right)\\ &-\sigma^2[\boldsymbol{x} - \boldsymbol{\psi}(\boldsymbol{y})]_a \operatorname{Tr}\left(U^\top \operatorname{Hess}_{\boldsymbol{y}}[\boldsymbol{\psi}_a]U\right) + \mathbb{E}_{\boldsymbol{\epsilon}} r(\sigma U\boldsymbol{\epsilon}),\end{aligned}$$

where we used that $\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{\epsilon}^\top M\boldsymbol{\epsilon}] = \operatorname{Tr}[M]$ for any symmetric matrix M since $\mathbb{E}_{\boldsymbol{\epsilon}}[\boldsymbol{\epsilon}_i\boldsymbol{\epsilon}_j] = \delta_{ij}$. Next, denote $R_a(\sigma) = \mathbb{E}_{\boldsymbol{\epsilon}} r(\sigma U\boldsymbol{\epsilon})$ we can use the Jensen inequality and subsequently (10)

$$|R_a(\sigma)| = |\mathbb{E}_{\boldsymbol{\epsilon}} r(\sigma U\boldsymbol{\epsilon})| \leq \mathbb{E}_{\boldsymbol{\epsilon}}|r(\sigma U\boldsymbol{\epsilon})| \leq \frac{L_a}{6}\mathbb{E}_{\boldsymbol{\epsilon}}\|\sigma U\boldsymbol{\epsilon}\|_2^3.$$

Next, we notice that $\|\sigma U\boldsymbol{\epsilon}\|_2 = \sigma(\boldsymbol{\epsilon}^\top U^\top U\boldsymbol{\epsilon})^{1/2} = \sigma\|\boldsymbol{\epsilon}\|_2$. It is useful to notice that $\|\boldsymbol{\epsilon}\|_2$ is distributed according to a chi distribution. By using this remark, we find

$$|R_a(\sigma)| \leq \sigma^3\frac{L_a}{6}\mathbb{E}_{\boldsymbol{\epsilon}}\|\boldsymbol{\epsilon}\|_2^3 = \sigma^3\frac{L_a}{6}\frac{\sqrt{2}(m+1)\Gamma((m+1)/2)}{\Gamma(m/2)},$$

where the last equality uses the expression for the third moment of the chi distribution and where the Gamma function Γ is the extension of the factorial to the complex numbers.

B Details on Evidence Lower Bound for St-RKM model

Now we discuss the details of ELBO given in section 4. The first term in (9) is

$$\begin{aligned}\mathbb{E}_{q_U(\boldsymbol{z}|\boldsymbol{x}_i)}[\log(p(\boldsymbol{x}_i|\boldsymbol{z}))] =& -\frac{1}{2\sigma_0^2}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbb{I})}\|\boldsymbol{x}_i - \boldsymbol{\psi}_{\boldsymbol{\xi}}(\mathbb{P}_U\boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}_i) + \sigma\mathbb{P}_U\boldsymbol{\epsilon} + \delta\mathbb{P}_{U^\perp}\boldsymbol{\epsilon})\|_2^2\\ &-\frac{d}{2}\log(2\pi\sigma_0^2),\end{aligned}$$

Table 2: Datasets and hyperparameters used for the experiments. N is the number of training samples, d the input dimension (resized images), m the subspace dimension and M the minibatch size.

Dataset	N	d	m	M
MNIST	60000	28×28	10	256
fMNIST	60000	28×28	10	256
SVHN	73257	$32 \times 32 \times 3$	10	256
Dsprites	737280	64×64	5	256
3Dshapes	480000	$64 \times 64 \times 3$	6	256
Cars3D	17664	$64 \times 64 \times 3$	3	256

where we used the following reparameterization following Kingma & Welling (2014): $\mathbb{E}_{q_U(\mathbf{z}|\boldsymbol{x}_i)}[f(\mathbf{z})] = \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbb{I})}\left[f\left(\mathbb{P}_U\boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}) + (\sigma\mathbb{P}_U + \delta\mathbb{P}_{U^\perp})\epsilon\right)\right]$, with $p(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\psi}_{\boldsymbol{\xi}}(\boldsymbol{z}), \sigma_0^2\mathbb{I})$ and $q_U(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}|\mathbb{P}_U\boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}), \sigma^2\mathbb{P}_U + \delta^2\mathbb{P}_{U^\perp})$. Clearly, the above expectation can be written as follows

$$\mathbb{E}_{\boldsymbol{\epsilon}}\mathbb{E}_{\boldsymbol{\epsilon}_\perp}\|\boldsymbol{x}_i - \boldsymbol{\psi}_{\boldsymbol{\xi}}(\mathbb{P}_U\boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}_i) + \sigma U\boldsymbol{\epsilon} + \delta U_\perp\boldsymbol{\epsilon}_\perp)\|_2^2,$$

with $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbb{I}_m)$ and $\boldsymbol{\epsilon}_\perp \sim \mathcal{N}(0, \mathbb{I}_{\ell-m})$. Hence, we fix $\sigma_0^2 = 1/2$ and take $\delta > 0$ to a numerically small value. For the other terms of (9), we use the formula giving the KL divergence between multivariate normals. Let $\mathcal{N}_0$ and $\mathcal{N}_1$ be ℓ-variate normal distributions with mean μ_0, μ_1 and covariance Σ_0, Σ_1 respectively. Then,

$$\mathrm{KL}(\mathcal{N}_0, \mathcal{N}_1) = \frac{1}{2}\left\{ \mathrm{Tr}(\Sigma_1^{-1}\Sigma_0) + (\mu_1 - \mu_0)^\top\Sigma_1^{-1}(\mu_1 - \mu_0) - \ell + \log\left(\frac{\det\Sigma_1}{\det\Sigma_0}\right) \right\}$$

By using this identity, we find the second term of (9):

$$\mathrm{KL}[q_U(\boldsymbol{z}|\boldsymbol{x}_i), q(\boldsymbol{z}|\boldsymbol{x}_i)] = \frac{1}{2}\Big\{\frac{m\sigma^2 + (\ell - m)\delta^2}{\gamma^2} + \frac{1}{\gamma^2}\|\boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - \mathbb{P}_U\boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)\|_2^2 - \ell + \log\big(\frac{\gamma^{2\ell}}{\sigma^{2m}\delta^{2(\ell-m)}}\big)\Big\},$$

where $q(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}|\boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}), \gamma^2\mathbb{I}_\ell)$. For the third term in (9), we find

$$\mathrm{KL}[q_U(\boldsymbol{z}|\boldsymbol{x}_i), p(\boldsymbol{z})] = \frac{1}{2}\Big\{\mathrm{Tr}((\sigma^2\mathbb{P}_U + \delta^2\mathbb{P}_{U^\perp})\Sigma^{-1}) + (\mathbb{P}_U\boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}_i))^\top\Sigma^{-1}(\mathbb{P}_U\boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)) + \log\det(\Sigma) - \ell - \log(\sigma^{2m}\delta^{2(\ell-m)})\Big\},$$

with $p(\boldsymbol{z}) = \mathcal{N}(0, \Sigma)$. By averaging over $i = 1, \ldots, n$, we obtain

$$\frac{1}{n}\sum_{i=1}^{n}\mathrm{KL}[q_U(\boldsymbol{z}|\boldsymbol{x}_i), p(\boldsymbol{z})] = \frac{1}{2}\Big\{\mathrm{Tr}((\sigma^2\mathbb{P}_U + \delta^2\mathbb{P}_{U^\perp})\Sigma^{-1}) + \mathrm{Tr}(\mathbb{P}_U C_{\boldsymbol{\theta}}\mathbb{P}_U\Sigma^{-1}) + \log\det(\Sigma) - \ell - \log(\sigma^{2m}\delta^{2(\ell-m)})\Big\},$$

where we used the cyclic property of the trace and $C_{\boldsymbol{\theta}} = \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)\boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)^\top$. This proves the analogous expression in section 4. Finally, the estimation of the optimal Σ can be done in parallel to the Maximum Likelihood Estimation of the covariance matrix of a multivariate normal.

C Datasets and Hyperparameters

We refer to Table 2 and Table 3 for specific details on model architectures, datasets and hyperparameters used in this paper. All models were trained on full-datasets and for maximum 1000 epochs. Further all datasets are scaled between [0-1] and are resized to 28×28 dimensions except Dsprites and Cars3D. The PyTorch library (single precision) in Python was used as the programming language on 8GB

Table 3: Model architectures. All convolutions and transposed-convolutions are with stride 2 and padding 1. Unless stated otherwise, layers have Parametric-RELU ($\alpha = 0.2$) activation functions, except output layers of the pre-image maps which have sigmoid activation functions (since input data is normalized $[0, 1]$). Adam and CayleyAdam optimizers have learning rates 2×10^{-4} and 10^{-4} respectively. Pre-image map/decoder network is always taken as transposed of feature map/encoder network. $c = 48$ for Cars3D; and $c = 64$ for all others. Further, $\hat{k} = 3$ and stride 1 for MNIST, fMNIST, SVHN and 3Dshapes; and $\hat{k} = 4$ for others. SVHN and 3Dshapes are resized to 28×28 input dimensions.

Dataset	**Architecture**
MNIST/fMNIST/ /SVHN/3Dshapes/ Dsprites/Cars3D	$\phi_\theta(\cdot) = \begin{cases} \mathit{Conv}\ [c] \times 4 \times 4; \\ \mathit{Conv}\ [c \times 2] \times 4 \times 4; \\ \mathit{Conv}\ [c \times 4] \times \hat{k} \times \hat{k}; \\ \mathit{FC}\ 256; \\ \mathit{FC}\ 50\ (\mathit{Linear}) \end{cases}$ $\psi_\zeta(\cdot) = \begin{cases} \mathit{FC}\ 256; \\ \mathit{FC}\ [c \times 4] \times \hat{k} \times \hat{k}; \\ \mathit{Conv}\ [c \times 2] \times 4 \times 4; \\ \mathit{Conv}\ [c] \times 4 \times 4; \\ \mathit{Conv}\ [c]\ (\mathit{Sigmoid}) \end{cases}$

NVIDIA QUADRO P4000 GPU. See Algorithm 1 for training the St-RKM model. In the case of FactorVAE, the discriminator architecture is same as proposed in the original paper (Kim & Mnih, 2018).

Disentanglement Metrics: MIG was originally proposed by Chen et al. (2018), however we use the modified metric as proposed in Locatello et al. (2019). Further we evaluate this score on 5000 test points across all the considered datasets. SAP and Eastwood's metrics both uses different classifiers to compute the importance of each dimension of the learned representation for predicting a ground-truth factor. For these metrics, we randomly sample 5000 and 3000 training and testing points respectively. To compute these metrics, we use the open source library available at github.com/google-research/disentanglement_lib.

D Ablation studies

Significance of the KPCA loss: In this section, we show an ablation study on the KPCA loss and evaluate its effect on disentanglement. We repeat the experiments of Section 5 on the *mini*-3DShapes dataset (floor hue, wall hue, object hue and scale: 8000 samples), where we consider 3 different variants of the proposed model:

1. **St-RKM** ($\sigma = 0$): The KPCA loss is optimized in a stochastic manner using the Cayley Adam optimizer, as proposed in the paper.
2. **Gen-RKM:** The KPCA loss is optimized exactly at each step by performing an eigendecomposition in each mini-batch (this corresponds to the algorithm in Pandey et al. (2021)).
3. **AE-PCA:** A standard AE is used and a reconstruction loss is minimized for the training. As a post-processing step, a PCA is performed on the latent embedding of the training data.

The encoder/decoder maps are the same across all the models, and for the AE-PCA model, additional linear layers are used to map the latent space to the subspace. From Table 4, we conclude that optimizing the KPCA loss during training improves disentanglement. Moreover, using a stochastic algorithm improves computation time and scalability with only a slight decrease in disentanglement score. Note that calculating the exact eigendecomposition at each step (Gen-RKM) comes with numerical difficulties. In particular, double floating-point precision has to be used together with a careful selection of the number of principal components to avoid ill-conditioned kernel matrices. This problem is not encountered when using the St-RKM training algorithm.

Smaller encoder/decoder architecture: In this section, we analyze the impact of the encoder/decoder architecture on the generation quality of considered models. The generation quality experiment of Section 5 is repeated on the fMNIST and MNIST dataset, where the architecture and hyperparameters are adapted from Dupont (2018). From Table 5 and Figure 9, we see that the overall FID scores and generation quality have improved, however, the relative scores among the models did not change significantly.

Analysis of St-RKM with a fixed U : We discuss here the role of the optimization of $\mathrm{St}(\ell, m)$ on disentanglement in the case of a classical AE loss ($\sigma = 0$). To do so, a matrix $\tilde{U} \in \mathrm{St}(\ell, m)$ is generated randomly[3] and kept fixed during the training of the following optimization problem

$$\min_{\boldsymbol{\theta},\boldsymbol{\xi}} \lambda \frac{1}{n}\sum_{i=1}^{n} L^{(0)}_{\boldsymbol{\xi},\tilde{U}}\left(\boldsymbol{x}_i, \boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)\right) + \underbrace{\frac{1}{n}\sum_{i=1}^{n} \|\mathbb{P}^{(\varepsilon)}_{\tilde{U}^{\perp}} \boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)\|_2^2}_{\text{regularized PCA objective}}, \tag{11}$$

with $\lambda = 1$ and where $\varepsilon \geq 0$ is a regularization constant and where the regularized

[3]Using a random $\tilde{U} \in \mathrm{St}(\ell, m)$ can be interpreted as sketching the encoder map in the spirit of Randomized Orthogonal Systems (ROS) sketches (see Yang et al. (2017)).

(or mollified) projector $\mathbb{P}^{(\varepsilon)}_{\tilde{U}^\perp} = \varepsilon(\tilde{U}\tilde{U}^\top + \varepsilon\mathbb{I}_\ell)^{-1}$ is used in order to prevent numerical instabilities. Indeed, if $\varepsilon = 0$, the second term in (11) (PCA term) is not strictly convex as a function of $\boldsymbol{\phi_\theta}$, since this quadratic form has flat directions along the column subspace of $\tilde{U}$. Our numerical simulations in single-precision PyTorch with $\varepsilon = 0$ exhibit instabilities, i.e., the PCA term in (11) takes negative values during the training. Hence, the regularized projector is introduced so that the PCA quadratic is strongly convex for $\varepsilon > 0$. This instability is not observed in the training of (5) where U is not fixed. This is one asset of our training procedure using optimization over Stiefel manifold. Explicitly, the regularized projector satisfies the following properties

- $\mathbb{P}^{(\varepsilon)}_{\tilde{U}^\perp} u_\perp = u_\perp$ for all $u_\perp \in (\text{range}(U))^\perp$,
- $\mathbb{P}^{(\varepsilon)}_{\tilde{U}^\perp} u = \varepsilon u$ for all $u \in \text{range}(U)$.

Thanks to the push-through identity, we have the alternative expression $\mathbb{P}^{(\varepsilon)}_{\tilde{U}^\perp} = \mathbb{I} - U(U^\top U + \varepsilon\mathbb{I}_m)^{-1}U^\top$. Therefore, it holds $\lim_{\varepsilon\to 0} \mathbb{P}^{(\varepsilon)}_{\tilde{U}^\perp} = \mathbb{P}_{\tilde{U}^\perp}$, as it should. In our experiments, we set $\varepsilon = 10^{-5}$. If $\varepsilon \leq 10^{-6}$, the regularized PCA objective in (11) takes negative values after a few epochs due to the numerical instability as mentioned above.

In Figure 10a, the evolution of the training objective (11) is displayed. It can be seen that the final objective has a lower value [$\exp(6.78) \approx 881$] when U is optimized compared to its fixed counterpart [$\exp(6.81) \approx 905$] showing the merit of optimizing over Stiefel manifold for the same parameter ε. Hence the subspace determined by range(U) has to be adapted to the encoder and decoder networks.

In other words, the training over $\boldsymbol{\theta}, \boldsymbol{\xi}$ is not sufficient to minimize the $\mathrm{St}(\ell, m)$ objective with Adam. Figure 10b further explores the latent traversals in the context of this ablation study. In the top row of Figure 10b (latent traversal in the direction of $\boldsymbol{u}_1$), both the shape of the object and the wall hue are changing. A coupling between wall hue and shape is also visible in the bottom row of this figure.

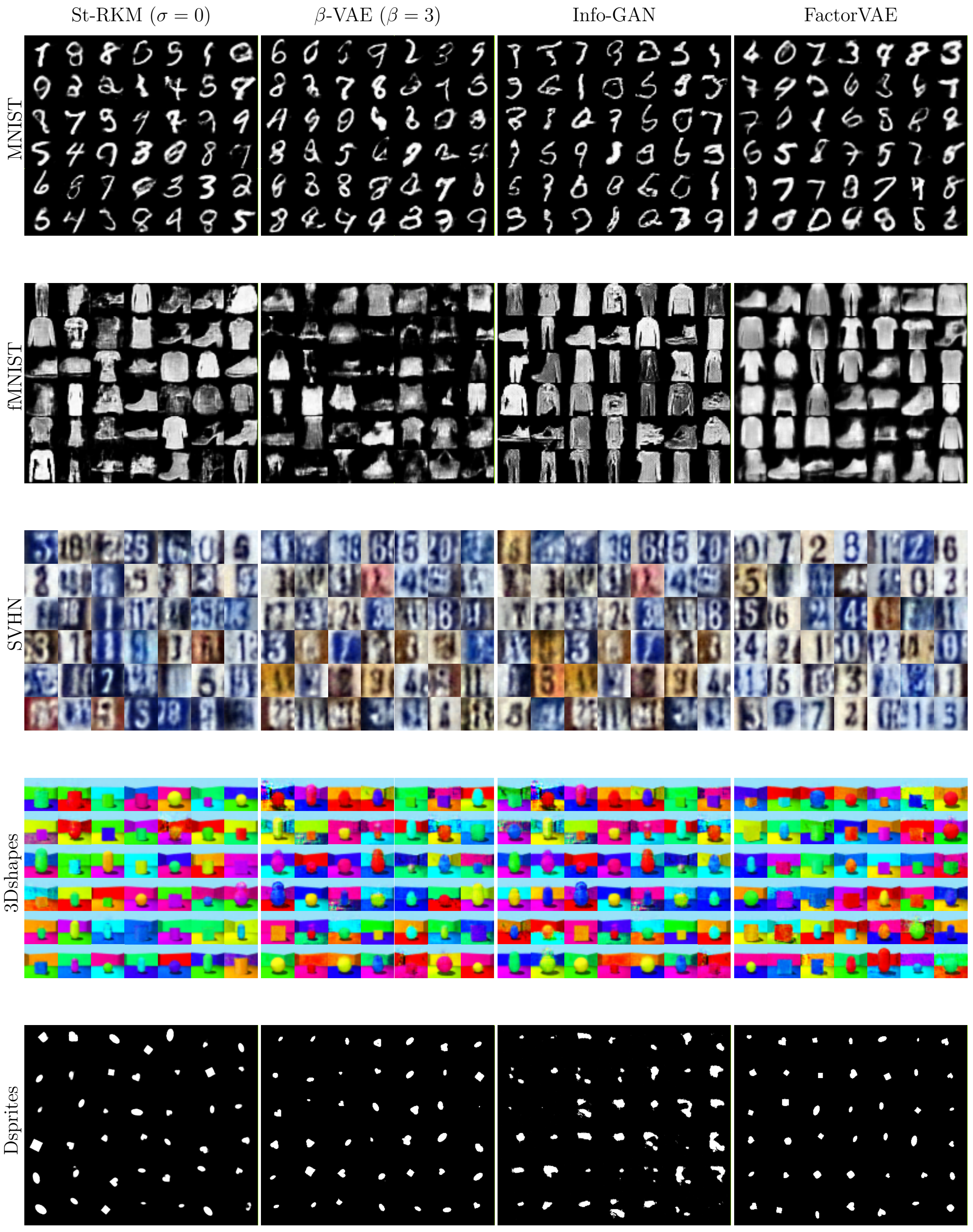


Figure 8: Samples of randomly generated batch of images used to compute FID scores and SWD scores (see Fig. 4).

Table 4: Training timings per epoch (in minutes) and disentanglement scores (Heusel et al. (2017)) for different variants of RKM when trained on the *mini*-3Dshapes dataset. Gen-RKM has the worst training time but gets the highest disentanglement scores. This is due to the exact eigendecomposition of the kernel matrix at every iteration. This computationally expensive step is approximated by the St-RKM model which achieves significant speed-up and scalability to large datasets. Finally, the AE-PCA model has the fastest training time due to the absence of eigendecompositions in the training loop. However, using PCA in the post-processing step alters the basis of the latent-space. This basis is unknown to the decoder network resulting in degraded disentanglement performance.

		St-**RKM** ($\sigma = 0$)	**Gen-RKM**	**AE-PCA**
Train. time		3.01 (0.71)	9.21 (0.54)	**2.87** (0.33)
Disent. score	Lasso	0.40 (0.02)	**0.44** (0.01)	0.35 (0.01)
	RF	0.27 (0.01)	**0.31** (0.02)	0.22 (0.02)
Compl. score	Lasso	**0.64** (0.01)	0.51 (0.01)	0.42 (0.01)
	RF	**0.67** (0.02)	0.58 (0.01)	0.45 (0.02)
Info. score	Lasso	**1.01** (0.02)	1.11 (0.02)	1.20 (0.01)
	RF	**0.98** (0.01)	1.09 (0.01)	1.17 (0.02)

Table 5: FID scores (lower is better, with standard deviations) computed on randomly generated 8000 images when trained with architecture and hyperparameters adapted from Dupont (2018).

	St-**RKM**	**VAE**	β-**VAE**	**FactorVAE**	**InfoGAN**
MNIST	**24.63** (0.22)	36.11 (1.01)	42.81 (2.01)	35.48 (0.07)	45.74 (2.93)
fMNIST	**61.44** (1.02)	73.47 (0.73)	75.21 (1.11)	69.73 (1.54)	84.11 (2.58)

Table 6: Computing the diagonalization scores (see Figure 3). Denote $M = \frac{1}{|\mathcal{C}|}\sum_{i\in\mathcal{C}} U_\star^\top \nabla\boldsymbol{\psi}(\boldsymbol{y}_i)\nabla\boldsymbol{\psi}(\boldsymbol{y}_i)^\top U_\star$, with $\boldsymbol{y}_i = \mathbb{P}_U \boldsymbol{\phi}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$ (cf. (8)). Then we compute the score as $\|M - \mathrm{diag}(M)\|_F \,/\, \|M\|_F$, where $\mathrm{diag}\colon \mathbb{R}^{m\times m} \mapsto \mathbb{R}^{m\times m}$ sets the off-diagonal elements of matrix to zero. The scores are computed for each model over 10 random seeds and show the mean (standard deviation). Lower scores indicate better diagonalization.

Models	**Dsprites**	**3Dshapes**	**Cars3D**
St-RKM-sl ($\sigma = 10^{-3}$, $U_\star$)	**0.17** (0.05)	**0.23** (0.03)	**0.21** (0.04)
St-RKM ($\sigma = 10^{-3}$, $U_\star$)	0.26 (0.05)	0.30 (0.10)	0.31 (0.09)
St-RKM ($\sigma = 10^{-3}$, random U)	0.61 (0.02)	0.72 (0.01)	0.69 (0.03)

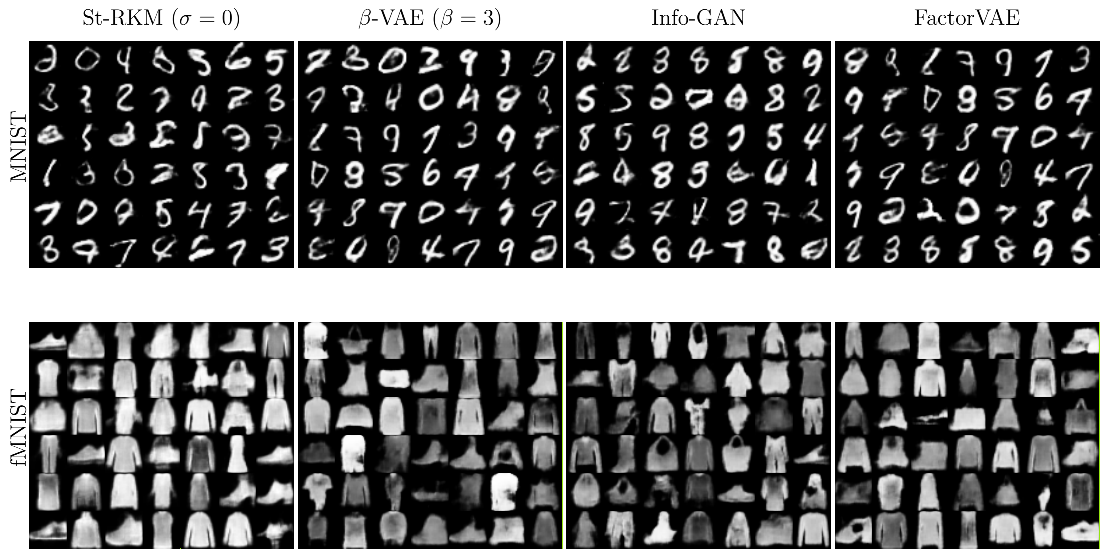
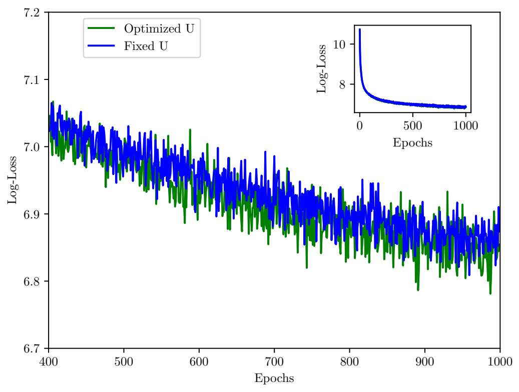
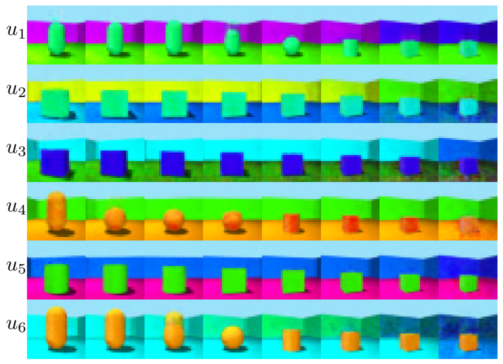


Figure 9: Samples of randomly generated images used to compute the FID scores; see Table 5.



(a) Loss-evolution



(b) Latent space Traversal

Figure 10: (a) Loss evolution (log plot) during the training of (11) over 1000 epochs with $\varepsilon = 10^{-5}$ once with Cayley-Adam optimizer (green curve) and then without (blue curve). (b) Traversals along the principal components when the model was trained with a fixed U, i.e. with the objective given by (11) and $\varepsilon = 10^{-5}$. There is no clear isolation of a feature along any of the principal components indicating further that optimizing over U is key for better disentanglement.

References

Absil, P.-A., Mahony, R., & Sepulchre, R. (2008). *Optimization Algorithms on Matrix Manifolds*. Princeton, NJ: Princeton University Press.

Avron, H., Nguyen, H., & Woodruff, D. (2014). Subspace Embeddings for the Polynomial Kernel. In *Advances in neural information processing systems* (Vol. 27, pp. 2258–2266).

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(8), 1798–1828.

Burgess, C., & Kim, H. (2018). *3Dshapes Dataset.* https://github.com/deepmind/3dshapes-dataset/.

Burgess, C. P., Higgins, I., Pal, A., Matthey, L., Watters, N., Desjardins, G., & Lerchner, A. (2017). Understanding disentangling in β-VAE. In *NIPS 2017 Workshop on Learning Disentangled Representations: from Perception to Control.*

Chen, R. T. Q., Li, X., Grosse, R. B., & Duvenaud, D. K. (2018). Isolating Sources of Disentanglement in Variational Autoencoders. In *Advances in neural information processing systems 31* (pp. 2610–2620).

Dupont, E. (2018). Learning disentangled joint continuous and discrete representations. In *Proceedings of the 32nd international conference on neural information processing systems* (pp. 708–718).

Eastwood, C., & Williams, C. K. I. (2018). A Framework for the Quantitative Evaluation of Disentangled Representations. In *proceedings of the International Conference on Learning Representations (ICLR).*

Ghosh, P., Sajjadi, M. S., Vergari, A., Black, M., & Schölkopf, B. (2020). From Variational to Deterministic Autoencoders. In *proceedings of the International Conference on Learning Representations (ICLR).*

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). GANs Trained by a Two Time-scale Update Rule Converge to a Local Nash Equilibrium. In *Advances in neural information processing systems* (pp. 6629–6640).

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., ... Lerchner, A. (2017). Beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In *proceedings of the International Conference on Learning Representations (ICLR)* (Vol. 2, p. 6).

Hinton, G. E. (2005). What kind of a graphical model is the brain? In *Proceedings of the 19th international joint conference on artificial intelligence* (p. 1765–1775). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Karras, T., Aila, T., Laine, S., & Lehtinen, J. (2017). Progressive Growing of GANs for Improved Quality, Stability, and Variation. In *proceedings of the international conference on learning representations (iclr).*

Kim, H., & Mnih, A. (2018). Disentangling by Factorising. In *proceedings of the thirty-fifth international conference on machine learning (icml)* (Vol. 80, pp. 2649–2658).

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

Kumar, A., Sattigeri, P., & Balakrishnan, A. (2018). Variational Inference of Disentangled Latent Concepts From Unlabeled Observations. In *International conference on learning representations.* Retrieved from `https://openreview.net/forum?id=H1kG7GZAW`

LeCun, Y., & Cortes, C. (2010). *MNIST handwritten digit database.* http://yann.lecun.com/exdb/mnist/.

LeCun, Y., Huang, F. J., & Bottou, L. (2004). Learning Methods for Generic Object Recognition with Invariance to Pose and Lighting. In *Computer Vision and Pattern Recognition (CVPR).*

Li, J., Li, F., & Todorovic, S. (2020). Efficient Riemannian Optimization on the Stiefel Manifold via the Cayley Transform. In *proceedings of the International Conference on Learning Representations (ICLR).*

Locatello, F., Bauer, S., Lučić, M., Rätsch, G., Gelly, S., Schölkopf, B., & Bachem, O. F. (2019). Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations. In *International conference on machine learning (icml).*

Locatello, F., Tschannen, M., Bauer, S., Rätsch, G., Schölkopf, B., & Bachem, O. (2020). Disentangling Factors of Variations Using Few Labels. In *International Conference on Learning Representations (ICLR).*

Matthey, L., Higgins, I., Hassabis, D., & Lerchner, A. (2017). *dsprites: Disentanglement testing sprites dataset.* https://github.com/deepmind/dsprites-dataset/.

Nesterov, Y. (2014). *Introductory lectures on convex optimization: A basic course* (1st ed.). Springer Publishing Company, Incorporated.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., & Ng, A. Y. (2011). Reading Digits in Natural Images with Unsupervised Feature Learning. In *Nips workshop on deep learning and unsupervised feature learning.*

Pandey, A., Schreurs, J., & Suykens, J. A. (2021). Generative restricted kernel machines: A framework for multi-view generation and disentangled feature learning. *Neural Networks*, *135*, 177 - 191.

Pandey, A., Schreurs, J., & Suykens, J. A. K. (2020). Robust Generative Restricted Kernel Machines using weighted conjugate feature duality. In *proceedings of the Sixth International Conference on Machine Learning, Optimization, and Data Science (LOD).*

Reed, S., Zhang, Y., Zhang, Y., & Lee, H. (2015). Deep Visual Analogy-Making. In *Advances in neural information processing systems.*

Rezende, D. J., & Mohamed, S. (2015). Variational Inference with Normalizing Flows. *International Conference on Machine Learning (ICML).*

Rolínek, M., Zietlow, D., & Martius, G. (2019). Variational Autoencoders pursue PCA directions (by accident). In *2019 ieee/cvf conference on computer vision and pattern recognition (cvpr)* (p. 12398-12407).

Salakhutdinov, R., & Hinton, G. (2009). Deep Boltzmann Machines. In *proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics* (Vol. 5 of JMLR).

Suykens, J. A. K. (2017, August). Deep Restricted Kernel Machines using conjugate feature duality. *Neural Computation*, *29*(8), 2123-2163.

Xiao, H., Rasul, K., & Vollgraf, R. (2017). Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv:1708.07747*.

Yang, Y., Pilanci, M., & Wainwright, M. J. (2017). Randomized sketches for kernels: Fast and optimal nonparametric regression. *The Annals of Statistics*, *45*(3), 991–1023.