GD-VAEs: Geometric Dynamic Variational Autoencoders for Learning Nonlinear Dynamics and Dimension Reductions

Ryan Lopez
Department of Physics
College of Creative Studies (CCS)
University of California Santa Barbara
Santa Barbara, CA 93106, USA

Paul J. Atzberger   ATZBERG@GMAIL.COM
Department of Mathematics
Department of Mechanical Engineering
University of California Santa Barbara
Santa Barbara, CA 93106, USA
http://atzberger.org

Abstract

We develop data-driven methods incorporating geometric and topological information to learn parsimonious representations of nonlinear dynamics from observations. The approaches learn nonlinear state-space models of the dynamics for general manifold latent spaces using training strategies related to Variational Autoencoders (VAEs). Our methods are referred to as Geometric Dynamic (GD) Variational Autoencoders (GD-VAEs). We learn encoders and decoders for the system states and evolution based on deep neural network architectures that include general Multilayer Perceptrons (MLPs), Convolutional Neural Networks (CNNs), and other architectures. Motivated by problems arising in parameterized PDEs and physics, we investigate the performance of our methods on tasks for learning reduced dimensional representations of the nonlinear Burgers Equations, Constrained Mechanical Systems, and spatial fields of Reaction-Diffusion Systems. GD-VAEs provide methods that can be used to obtain representations in manifold latent spaces for diverse learning tasks involving dynamics.

Keywords: variational autoencoders, dimension reduction, dynamical systems, scientific machine learning

Introduction

We develop data-driven approaches for learning predictive models and representations from observations of dynamical processes. We introduce learning methods that incorporate inductive biases leveraging known prior topological and geometric information. This allows for using insights from techniques in the qualitative analysis of dynamical systems and other domain knowledge. In practice, the observation data can include experimental measurements, large-scale computational simulations, or solutions of complex dynamical systems for which we seek reduced descriptions. Learning representations with prescribed target properties can be used to help enhance the robustness of predictions, yield more interpretable results, or provide further insights into the underlying mechanisms generating observed behaviors.

A central challenge in the dynamical setting is to learn informative representations facilitating stable predictions for multiple future states of the system [59]. For this purpose, we develop probabilistic autoencoders that incorporate noise-based regularizations and geometric priors to learn smooth reduced dimensional representations for the observations. We train our probabilistic autoencoders building on the framework of Variational Autoencoders (VAEs) [45]. To incorporate known topological and geometric information, we develop learning methods allowing for general manifold latent spaces. This presents challenges to obtain neural networks to represent mappings

to the manifold geometry and for gradient-based training. We address these issues by developing mappings based on extrinsic geometric descriptions and by developing custom back-propagation methods for passage of gradient information through our manifold latent spaces. We demonstrate these methods for learning representations and performing prediction for constrained mechanical systems, parameterized non-linear PDES, Burger's equation, and Reaction-Diffusion systems.

The general problem of learning dynamical models from a time series of observations has a long history spanning many fields [59; 76; 18; 38]. This includes the fields of dynamical systems [10; 76; 78; 55; 58; 60; 34; 21; 25], control [11; 59; 70; 74], statistics [1; 56; 28], and machine learning [18; 38; 51; 68; 5; 86; 52]. Many of the most successful and widely-used approaches rely on assumptions on the model structure, most commonly, that a time-invariant linear dynamical systems (LDS) provides a good local approximation or that the noise is Gaussian. These include the Kalman Filter and extensions [43; 24; 30; 82; 84], Proper Orthogonal Decomposition (POD) [15; 57], and more recently Dynamic Mode Decomposition (DMD) [74; 49; 81] and Koopman Operator approaches [58; 22; 47]. Methods for learning nonlinear dynamics include the NARX and NOE approaches with function approximators based on neural networks and other models classes [59; 76], sparse symbolic dictionary methods that are linear-in-parameters using LASSO [71; 79] such as SINDy [11; 75; 76], and dynamic Bayesian networks (DBNs), such as Hidden Markov Chains (HMMs) and Hidden-Physics Models [68; 62; 73; 7; 48; 28].

Many variants of autoencoders have been developed for making predictions of sequential data. This includes those based on Recurrent Neural Networks (RNNs) with LSTMs and GRUs [37; 31; 19]. Approaches for incorporating topological information into latent variable representations include the early works by Kohonen on Self-Organizing Maps (SOMs) [46] and Bishop on Generative Topographical Maps (GTMs) based on density networks providing a generative approach [8]. While general RNNs provide a rich approximation class for sequential data, they pose for dynamical systems challenges for interpretability and for training to obtain predictions stable over many steps. Autoencoders have also been combined with symbolic dictionary learning for latent dynamics in [14] providing some advantages for interpretability and robustness. Dictionary methods however require specification in advance of sufficiently expressive libraries of functions. Neural networks incorporating physical information have also been developed, where some of the methods use regularizations introduced during training to enhance stability [61; 3; 80; 51; 80; 4; 26]. The work of [20] considers methods for processing of speech and handwriting by investigating RNNs combined with VAEs to obtain more robust sequential models.

In our work, we learn dynamical models building on the VAE framework to train probabilistic encoders and decoders between general manifold latent spaces. This provides additional regularizations and constraints to help promote parsimoniousness, disentanglement of features, robustness, and interpretability. Prior VAE methods used for dynamical systems include [33; 63; 52; 29; 16; 63; 69]. These works use primarily Euclidean latent spaces and consider applications such as human motion capture and ODE systems. More recently, VAE methods using non-Euclidean latent spaces include [41; 42; 17; 52; 2]. These incorporate the role of geometry by augmenting the prior distribution on a Euclidean latent space to bias encodings toward a manifold. In the recent works [65; 23; 27], explicit projection procedures are introduced to map analytically or to sample through random walks an embedded manifold. These works primarily consider geometries based on $SO(3)$, spheres, tori, and cylinders. We develop in our work more explicit procedures that can be used with back-propagation for more general geometries, including non-orientable manifolds. We also discuss related methods in our proceedings paper [53].

We introduce here further methods for more general latent space representations, including non-orientable manifolds, and applications to parameterized PDEs, constrained mechanical systems, and reaction-diffusion systems. We introduce general methods for non-Euclidean latent spaces in terms of point-cloud representations of the manifold along with local gradient information that can be utilized within general back-propagation frameworks. This allows for manifolds that have complex shapes or arise from other unsupervised learning approaches.

The paper is organized as follows. In Section 1, we formulate the probabilistic encoder-decoder models and use the Evidence Lower Bound (ELBO) to derive and motivate the loss functions and regularizations used for training. In Section 2, we discuss the challenges in learning with manifold latent spaces. We also develop methods for mapping to general manifolds and for performing back-propagation for gradient-based training. In Section 3, we present results demonstrating the methods for learning representations for constrained mechanical systems, parameterized non-linear PDEs, Burgers equation, and Reaction-Diffusion systems. To investigate the role of the non-linear mappings and noise regularizations in our approaches, we also make comparisons of our methods with analytic techniques and some other widely-used data-driven methods. We also present additional derivations in Appendix A and B. We present an additional result on how learned covariances can be used to help identify geometric structures in Appendix C. The introduced GD-VAE methods provide ways to incorporate topological and geometric information when learning representations for dynamical tasks.

1. Learning Nonlinear Dynamics with Variational Autoencoders (VAEs)

A central challenge in the non-linear setting is to learn from observations informative parsimonious representations of the dynamics. In practice, observation data can include experimental measurements, large-scale computational simulations, or solutions to more complicated dynamical systems for which we seek reduced descriptions. Such representations can then be used to make predictions, to perform simulations, or as part of optimization for designing controllers [59].

We develop data-driven approaches building on the Variational Autoencoder (VAE) framework [45]. In contrast to standard autoencoders, which can result in scattered disconnected encodings, VAEs train probabilistic encoders and decoders where noise provides additional regularizations. In VAEs this promotes smoother dependence on inputs, more connectedness between the encodings, and disentanglement of encoded components [45].
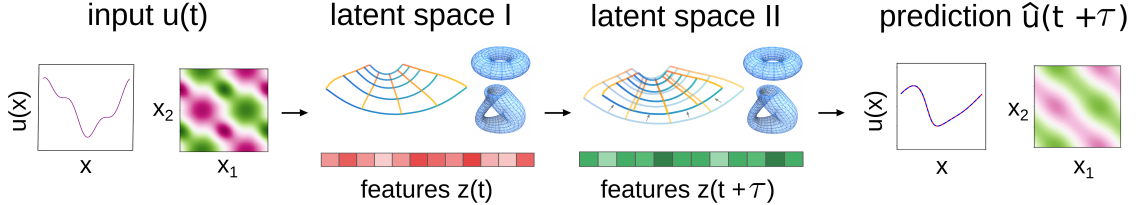

Figure 1: **Learning Nonlinear Dynamics with Encoders and Decoders.** Data-driven methods are developed for learning robust models to predict from $u(x,t)$ the non-linear evolution to future states $u(x, t + \tau)$ for PDEs and other dynamical systems. Probabilistic autoencoders are utilized to learn representations $\mathbf{z}$ of $u(x,t)$ in reduced dimensional latent spaces with prescribed geometric and topological properties. Our trainable models make predictions using learnable maps that (i) encode an input $u(x,t) \in \mathcal{U}$ as $\mathbf{z}(t)$ in latent space *(left)*, (ii) evolve the representation $\mathbf{z}(t) \to \mathbf{z}(t + \tau)$ *(middle)*, (iii) decode the representation $\mathbf{z}(t + \tau)$ to predict $\hat{u}(x, t + \tau)$ *(right)*.

We learn VAE predictors using a Maximum Likelihood Estimation (MLE) approach. For dynamic predictions $\mathbf{X} \to \mathbf{x}$, we use a Log Likelihood (LL) which can be expressed as

$$\mathcal{J}_{LL} = \log(p_\theta(\mathbf{X}, \mathbf{x})) = \underbrace{\log(p_\theta(\mathbf{x}|\mathbf{X}))}_{\mathcal{J}_{LC}} + \underbrace{\log(p_\theta(\mathbf{X}))}_{\mathcal{J}_{LX}}. \tag{1}$$

3

The $\mathcal{J}_{LC}$ gives the Log-Conditional (LC) Likelihood and $\mathcal{J}_{LX}$ gives the marginal Log Likelihood of **X**. For predicting the dynamics of $u(t)$, we discretize in time. We sample trajectories as $\{u(s_k)\}_{k=0}^{N}$, where $s_k = t_0 + k\tau$ and τ is the sampling time-scale. We learn representations for the discretized dynamics $\mathbf{X} \to \mathbf{x}$ by letting $\mathbf{X} = u(s_k)$ and $\mathbf{x} = u(s_{k+1})$. We base p_θ on the autoencoder framework in Figure 1 and 2. More specifically, we use variational inference to approximate the LL by the Evidence Lower Bound (ELBO) [9; 45]. In particular, we use the concavity of $\log(\cdot)$ and Jensen's Inequality for $\log \mathbb{E}[(\cdot)] \geq \mathbb{E}[\log(\cdot)]$. This yields the lower-bound

$$
\begin{aligned}
\mathcal{J}_{LL} &= \log(p(\mathbf{x}|\mathbf{X})p(\mathbf{X})) = \log \mathbb{E}_{q(\mathbf{z}|\mathbf{X})}\left[\frac{p(\mathbf{x}, \mathbf{z}|\mathbf{X})p(\mathbf{X})}{q(\mathbf{z}|\mathbf{X})}\right] = \log \mathbb{E}_{q(\mathbf{z}|\mathbf{X})}\left[\frac{p(\mathbf{x}|\mathbf{z})p(\mathbf{z}|\mathbf{X})p(\mathbf{X})}{q(\mathbf{z}|\mathbf{X})}\right] \\
&\geq \mathbb{E}_{q(\mathbf{z}|\mathbf{X})}\left[\log\left(\frac{p(\mathbf{x}|\mathbf{z})p(\mathbf{z}|\mathbf{X})p(\mathbf{X})}{q(\mathbf{z}|\mathbf{X})}\right)\right] = \mathbb{E}_{q(\mathbf{z}|\mathbf{X})}\left[\log\left(\frac{p(\mathbf{x}|\mathbf{z})p(\mathbf{X}|\mathbf{z})p(\mathbf{z})}{q(\mathbf{z}|\mathbf{X})}\right)\right] \qquad (2) \\
&= \underbrace{\mathbb{E}_{q(\mathbf{z}|\mathbf{X})}\left[\log\left(p(\mathbf{x}|\mathbf{z})\right)\right]}_{\mathcal{J}_{RE}} + \underbrace{\mathbb{E}_{q(\mathbf{z}|\mathbf{X})}\left[\log\left(p(\mathbf{X}|\mathbf{z})\right)\right]}_{\mathcal{J}_{RR}} + \underbrace{\mathbb{E}_{q(\mathbf{z}|\mathbf{X})}\left[\log\left(\frac{p(\mathbf{z})}{q(\mathbf{z}|\mathbf{X})}\right)\right]}_{\mathcal{J}_{KL}},
\end{aligned}
$$

$$\Rightarrow \mathcal{J}_{LL} \geq \mathcal{J}_{RE} + \mathcal{J}_{RR} + \mathcal{J}_{KL}.$$

To simplify the notation for the expressions above, we have suppressed dependence on the parameters θ. We assume here the considered probability distributions factor to have an autoencoding-like property so that $p(\mathbf{X}, \mathbf{x}, \mathbf{z}) := p(\mathbf{x}|\mathbf{z})p(\mathbf{z}|\mathbf{X})p(\mathbf{X})$. This ensures the encoding $\mathbf{z}$ captures all the relevant information to generate $\mathbf{x}$, so that $p(\mathbf{x}|\mathbf{z}, \mathbf{X}) = p(\mathbf{x}|\mathbf{z})$. The $q(\mathbf{x}|\mathbf{z})$ gives the distribution for a probabilistic encoder map from $\mathbf{x}$ to $\mathbf{z}$. The $p(\mathbf{x}|\mathbf{z})$ gives a probabilistic decoder map that reconstructs $\mathbf{x}$ from $\mathbf{z}$.

The encoding and decoding distributions in the variational approximations will often be taken to be learnable Gaussians of the form $p_\theta(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_d(\mathbf{z}; \theta), \boldsymbol{\Sigma}_d(\theta))$ and $q_\theta(\mathbf{z}|\mathbf{X}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_e(\mathbf{X}; \theta), \boldsymbol{\Sigma}_e(\theta))$. The $\mathcal{N}$ denotes the probability density of the multi-variate Gaussian with mean $\boldsymbol{\mu}(\mathbf{z}; \theta)$ and covariance $\boldsymbol{\Sigma}(\theta)$. In the case of no noise $\boldsymbol{\Sigma}_e = 0$, the $\boldsymbol{\mu}_e(\mathbf{X})$ would serve as a deterministic encoding map of $\mathbf{x}$ to obtain $\mathbf{z}$. When there is noise $\boldsymbol{\Sigma}_e \neq 0$, $\boldsymbol{\mu}_e$ becomes the center location of the Gaussian in latent space and is the most probable encoding of $\mathbf{x}$ to obtain $\mathbf{z}$, see Figure 2 *(lower-left)*. This holds similarly for the probabilistic decoder map given by $\boldsymbol{\mu}_d$ and $\boldsymbol{\Sigma}_d$. For the decoder, the log probability gives a weighted ℓ^2-norm penalizing errors in reconstructing $\mathbf{x}$, since for a Gaussian $-\log(p(\mathbf{x}|\mathbf{z})) = \frac{1}{2}\|\mathbf{x} - \boldsymbol{\mu}(\mathbf{z})\|_{\Sigma}^2 + c_0$. Here, the c_0 is a constant independent of $\mathbf{x}$ and

$$\|\mathbf{x} - \boldsymbol{\mu}(\mathbf{z})\|_{\Sigma}^2 = (\mathbf{x} - \boldsymbol{\mu}(\mathbf{z}))^T \Sigma^{-1}(\mathbf{x} - \boldsymbol{\mu}(\mathbf{z})). \qquad (3)$$

This holds similarly for $-\log(p(\mathbf{X}|\mathbf{z}))$ in reconstructing $\mathbf{X}$ from $\mathbf{z}$.

It is convenient to reformulate the maximization of the terms of the lower-bound in equation 2 to a minimization problem with loss function $\mathcal{L}_{LL} = -\mathcal{J}_{LL}$. The optimization then can be interpreted as minimizing the reconstruction losses

$$
\begin{aligned}
\mathcal{L}_{RE} &= -\mathcal{J}_{RE} = -\mathbb{E}_{q(\mathbf{z}|\mathbf{X})}\left[\log\left(p(\mathbf{x}|\mathbf{z})\right)\right] & (4) \\
\mathcal{L}_{RR} &= -\mathcal{J}_{RR} = -\mathbb{E}_{q(\mathbf{z}|\mathbf{X})}\left[\log\left(p(\mathbf{X}|\mathbf{z})\right)\right], & (5)
\end{aligned}
$$

subject to further regularization based on reducing the Kullback-Lieber (KL)-Divergence between $q(\mathbf{z}|\mathbf{X})$ and $p(\mathbf{z})$,

$$\mathcal{L}_{KL} = -\mathcal{J}_{KL} = -\mathbb{E}_{q(\mathbf{z}|\mathbf{X})}\left[\log\left(\frac{p(\mathbf{z})}{q(\mathbf{z}|\mathbf{X})}\right)\right] = \mathcal{D}_{KL}\left(q(\mathbf{z}|\mathbf{X})|p(\mathbf{z})\right). \qquad (6)$$

We also split the latent codes as $\bar{\mathbf{z}} = (\mathbf{z}, \mathbf{z}')$ and introduce the additional distribution $p_{\theta_\ell}(\mathbf{z}'|\mathbf{z})$ which models steps of the latent space dynamics $\mathbf{z}(t) \to \mathbf{z}(t + \tau)$ where $\mathbf{z} = \mathbf{z}(t)$ and $\mathbf{z}' = \mathbf{z}(t + \tau)$.

We restrict the decoder distributions to have dependence $p(\mathbf{X}|\bar{\mathbf{z}}) = p(\mathbf{X}|\mathbf{z})$ and $p(\mathbf{x}|\bar{\mathbf{z}}) = p(\mathbf{x}|\mathbf{z}')$. In the case of deterministic dynamics, we alternatively can express this by introducing the latent-space mapping $\mathbf{z}' = f_{\theta_\ell}(\mathbf{z})$ parameterized by θ_ℓ. We also augment the $\mathcal{D}_{KL}$ term to serve as an adjustable regularization weighted by the parameter β and also allow for adjusting the relative strength of the reconstructions for $\mathbf{x}$ and $\mathbf{X}$ using the parameter γ. Our models have the learnable parameters $\theta = (\theta_e, \theta_d, \theta_\ell)$ with θ_e for the encoder, θ_d for the decoder, and θ_ℓ for the dynamics.

In summary, we train our models based on minimizing the loss function $\mathcal{L}^B = \sum_{i=1}^m \mathcal{L}_i^B$ with

$$
\begin{aligned}
\theta^* &= \arg\min_{\theta \in \Theta} \mathcal{L}^B(\theta_e, \theta_d, \theta_\ell; \{\mathbf{X}^{(i)}, \mathbf{x}^{(i)}\}_{i=1}^m), \quad \mathcal{L}_i^B = \mathcal{L}_{RE}^{(i)} + \mathcal{L}_{KL}^{(i)} + \mathcal{L}_{RR}^{(i)}, &(7)\\
\mathcal{L}_{RE}^{(i)} &= -E_{\mathfrak{q}_{\theta_e}(\mathbf{z}|\mathbf{X}^{(i)})}\left[\log \mathfrak{p}_{\theta_d}(\mathbf{x}^{(i)}|\mathbf{z}')\right], \quad \mathcal{L}_{KL}^{(i)} = \beta \mathcal{D}_{KL}\left(\mathfrak{q}_{\theta_e}(\mathbf{z}|\mathbf{X}^{(i)}) \,\|\, \tilde{\mathfrak{p}}_{\theta_d}(\mathbf{z})\right), &\\
\mathcal{L}_{RR}^{(i)} &= -\gamma E_{\mathfrak{q}_{\theta_e}(\mathbf{z}|\mathbf{X}^{(i)})}\left[\log \mathfrak{p}_{\theta_d}(\mathbf{X}^{(i)}|\mathbf{z})\right].
\end{aligned}
$$

The $\mathfrak{q}_{\theta_e}$ denotes the distribution for the probabilistic encoding map, $\mathfrak{p}_{\theta_d}$ denotes the distribution of the probabilistic decoding map, and $\tilde{\mathfrak{p}}_{\theta_d}$ the prior distribution used for regularizing the organization of the latent-space representation $\bar{\mathbf{z}}$. This is trained using batches of m observation samples $\{(\mathbf{X}^{(i)}, \mathbf{x}^{(i)})\}_{i=1}^m$. For a dynamical system with a collection of sampled trajectories $\{u^k(s)\}_k$, we have that $\mathbf{X}^{(i)}$ samples the initial state $u^{k_i}(t_i)$ and $\mathbf{x}^{(i)}$ samples the future state $u^{k_i}(t_i + \tau)$, where k_i gives the trajectory index and t_i the sampled initial time.

The loss function in equation 7 provides a regularized form of MLE. The three terms comprising $\mathcal{L}^B$ can be interpreted as follows (i) $\mathcal{L}_{RR}$ for the log likelihood of reconstructing samples, (ii) $\mathcal{L}_{RE}$ is the log likelihood of correctly predicting samples after a single time step, and (iii) $\mathcal{L}_{KL}$ is a regularization term based on the Kullback-Liebler (KL) Divergence between the encoding distribution and latent space prior distribution.

For the trainable encoder and decoder probability distributions, we use parameterizations for the Gaussians of the form

$$
\begin{aligned}
\mathfrak{q}_{\theta_e}(\mathbf{z}|\mathbf{X}) &= \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_e(\mathbf{X}; \theta_e), \boldsymbol{\Sigma}_e(\mathbf{X}; \theta_e)) &(8)\\
\mathfrak{p}_{\theta_d}(\mathbf{X}|\mathbf{z}) &= \mathcal{N}(\mathbf{X}; \boldsymbol{\mu}_d(\mathbf{z}; \theta_d), \boldsymbol{\Sigma}_d(\mathbf{z}; \theta_d)) &(9)\\
\mathfrak{p}_{\theta_d}(\mathbf{x}|\mathbf{z}') &= \mathcal{N}(\mathbf{x}; \boldsymbol{\mu'}_d(\mathbf{z}'; \theta_d), \boldsymbol{\Sigma'}_d(\mathbf{z}'; \theta_d)). &(10)
\end{aligned}
$$

The $\mathcal{N}$ denotes the density of the multivariate Gaussian with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. We also can express this more directly in terms of the probabilistic maps and resulting random variables as

$$
\begin{aligned}
\mathbf{z} &\sim \mathfrak{q}_{\theta_e}(\mathbf{z}|\mathbf{X}), & \mathbf{z} &= \boldsymbol{\mu}_e(\mathbf{X}; \theta_e) + \boldsymbol{\eta}_e \\
\mathbf{z}' &\sim \mathfrak{p}_{\theta_\ell}(\mathbf{z}'|\mathbf{z}), & \mathbf{z}' &= f_{\theta_\ell}(\mathbf{z}) \\
\mathbf{x} &\sim \mathfrak{p}_{\theta_d}(\mathbf{x}|\mathbf{z}'), & \mathbf{x} &= \boldsymbol{\mu'}_d(\mathbf{z}'; \theta_d) + \boldsymbol{\eta'}_d \\
\mathbf{X} &\sim \mathfrak{p}_{\theta_d}(\mathbf{X}|\mathbf{z}), & \mathbf{X} &= \boldsymbol{\mu}_d(\mathbf{z}; \theta_d) + + \boldsymbol{\eta}_d.
\end{aligned} \quad (11)
$$

The $\boldsymbol{\eta}_*$ denote noise regularizations provided by Gaussian random variables with mean zero and covariance $\boldsymbol{\Sigma}_*$.

The terms that can be learned in this dynamic VAE framework are $(\boldsymbol{\mu}_e, \boldsymbol{\Sigma}_e, \boldsymbol{\mu}_d, \boldsymbol{\Sigma}_d, \boldsymbol{\mu'}_d, \boldsymbol{\Sigma'}_d, f_{\theta_\ell})$, which are parameterized by $\theta = (\theta_e, \theta_d, \theta_\ell)$. The $\boldsymbol{\mu}_e, \boldsymbol{\mu}_d, \boldsymbol{\mu'}_d$ denote learnable functions encoding or decoding the inputs. These can be represented using deep neural networks or other model classes, see Figure 2. The noise serves as an additional source of regularization to help promote smoothness and connectedness of the encodings, see Figure 2 *(lower-left)*. In practice, while the variances are learnable, for many problems it can be useful to treat the $\boldsymbol{\Sigma}_*$ as hyper-parameters. We discuss some ways to use learnable covariances $\boldsymbol{\Sigma}_*$ to extract from the data useful geometric structures in Appendix C.

We learn predictors for the dynamics by training over samples of evolution pairs $\{(u_i^{k_i}, u_{i+1}^{k_i})\}_{i=1}^m$, where i denotes the sample index and $u_i^{k_i} = u^{k_i}(s_i)$ with $s_i = t_0 + i\tau$ for a time-scale τ. To make
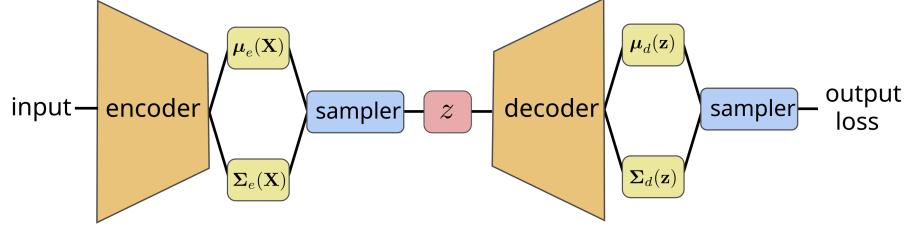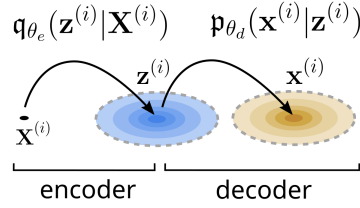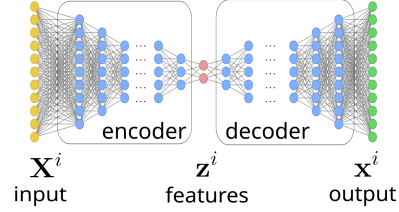
Figure 2: **Variational Autoencoder (VAE) Framework.** Representations of the non-linear dynamics are learned using probabilistic encoders and decoders trained using the VAE framework [45]. Our deep neural network (DNN) models have the VAE architecture shown at the *(top)*. This includes learnable probabilistic mappings, in contrast to more conventional neural network autoencoders shown on the *(bottom-right)*. GD-VAEs includes a probabilistic encoder with distribution $\mathfrak{q}_{\theta_e}$ and a probabilistic decoder with distribution $\mathfrak{p}_{\theta_d}$, shown on the *(bottom-left)*. These DNNs are trained (i) to serve as feature extractors to represent functions $u(x, t)$ and their evolution in a reduced dimensional latent space as $\mathbf{z}(t) \rightarrow \mathbf{z}(t+\tau)$, and (ii) to serve as approximators that can construct predictions $u(x, t+\tau)$ using the features $\mathbf{z}(t + \tau)$.

predictions, the learned models use the following stages: (i) extract from $u(t)$ the features $z(t)$, (ii) evolve $z(t) \rightarrow z(t + \tau)$, (iii) predict using $z(t + \tau)$ the $\hat{u}(t + \tau)$, see Figures 1 and 2. By function composition, the learned model also can make multi-step predictions for the dynamics.

2. Learning with Manifold Latent Spaces having General Geometries and Topologies

For many systems, parsimonious representations can be obtained by working with non-Euclidean manifold latent spaces. This includes tori for doubly periodic systems or even non-orientable manifolds, such as a klein bottle as arises in imaging and perception studies [13]. For this purpose, we learn encoders $\mathcal{E}$ over a family of mappings to a prescribed manifold $\mathcal{M}$ of the form

$$\mathbf{z} = \mathcal{E}_\phi(\mathbf{x}) = \Lambda(\tilde{\mathcal{E}}_\phi(\mathbf{x})) = \Lambda(\mathbf{w}), \text{ where } \mathbf{w} = \tilde{\mathcal{E}}_\phi(\mathbf{x}). \tag{12}$$

The $\mathcal{E}_\phi$ denotes a candidate encoder to the manifold when the parameters are ϕ. This can be used as part of the encoder map with $\boldsymbol{\mu}_e(\cdot) = \mathcal{E}_\phi(\cdot)$, see equation 11 and Figure 3.

To generate a family of maps over which we can learn in practice, we use that a smooth closed manifold $\mathcal{M}$ of dimension m can be embedded within $\mathbb{R}^{2m}$, as supported by the Whitney Embedding Theorem [85]. We obtain a family of maps to the manifold by constructing maps in two steps using
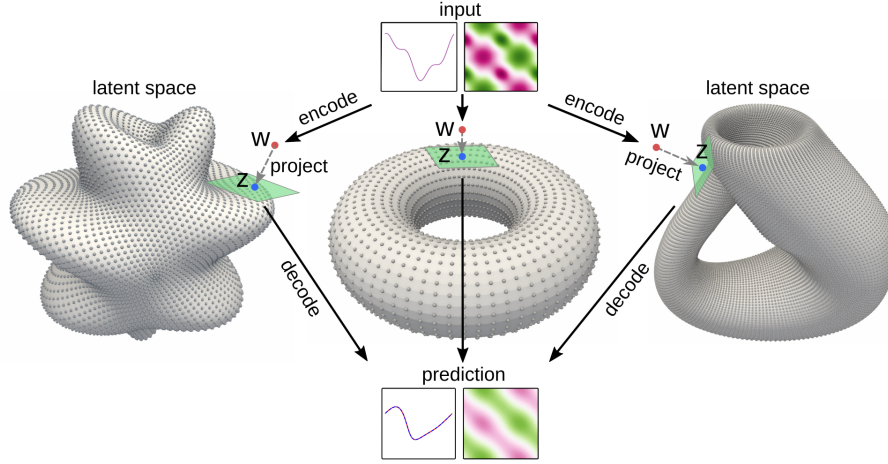
Figure 3: **Learnable Mappings for Manifold Latent Spaces.** We develop methods for using manifold latent space representations having general geometries and topologies through projection maps. For inter-operability with widely used trainable mappings, such as neural networks, we use a strategy of mapping inputs first to a point $\mathbf{w} = \tilde{\mathcal{E}}_\phi(\mathbf{x}) \in \mathbb{R}^N$ for the embedding space which is then projected to a point in the manifold $\mathbf{z} = \Lambda(\mathbf{w}) \in \mathcal{M} \subset \mathbb{R}^N$. This provides trainable mappings for general manifold latent spaces. In practice, we can represent the manifold and compute projections based on general point cloud representations, analytic descriptions, product spaces, or other approaches.

equation 12. In the first step, we use an unconstrained encoder $\tilde{\mathcal{E}}$ from $\mathbf{x}$ to a point $\mathbf{w}$ in the embedding space. In the second step, we use a map Λ that projects a point $\mathbf{w} \in \mathbb{R}^{2m}$ to a point $\mathbf{z} \in \mathcal{M} \subset \mathbb{R}^{2m}$ within the embedded manifold. In this way, $\tilde{\mathcal{E}}$ can be any learnable mapping from $\mathbb{R}^n$ to $\mathbb{R}^{2m}$ for which there are many model classes including neural networks. To obtain a particular manifold map, the $\tilde{\mathcal{E}}$ only needs to learn an equivalent mapping from $\mathbf{x}$ to $\mathbf{w}$, where $\mathbf{w}$ is in the appropriate equivalence class $\mathcal{Q}_{\mathbf{z}}$ of a target point $\mathbf{z}$ on the manifold, $\mathbf{w} \in \mathcal{Q}_{\mathbf{z}} = \{\mathbf{w} \mid \Lambda(\mathbf{w}) = \mathbf{z}\}$. Here, we accomplish this in practice two ways: (i) we provide an analytic mapping Λ to $\mathcal{M}$, (ii) we provide a high resolution point-cloud representation of the target manifold along with local gradients and use for Λ a quantized or interpolated mapping to the nearest point on $\mathcal{M}$. We provide more details of our methods below and in Appendix A.

In practice, we can view the projection map $\mathbf{z} = \Lambda(\mathbf{w})$ to the manifold as the solution of the optimization problem

$$\mathbf{z}^* = \arg \min_{\mathbf{z} \in \mathcal{M}} \frac{1}{2}\|\mathbf{w} - \mathbf{z}\|_2^2. \tag{13}$$

We can always express patches of a smooth manifold using local coordinate charts $\mathbf{z} = \sigma^k(\mathbf{u})$ for $\mathbf{u} \in \mathcal{U} \subset \mathbb{R}^m$. For example, we could use in practice a local Monge-Gauge quadratic fit to a point cloud representation of the manifold, as in [32]. We can express $\mathbf{z}^* = \sigma^{k^*}(\mathbf{u}^*)$ for some chart k^* for solution of equation 13. In terms of the collection of coordinate charts $\{\mathcal{U}^k\}$ and local parameterizations $\{\sigma^k(\mathbf{u})\}$, we can express this as

$$\mathbf{u}^*, k^* = \arg \min_{k, \mathbf{u} \in \mathcal{U}^k} \Phi_k(\mathbf{u}, \mathbf{w}), \text{ where } \Phi_k(\mathbf{u}, \mathbf{w}) = \frac{1}{2}\|\mathbf{w} - \sigma^k(\mathbf{u})\|_2^2. \tag{14}$$

The $\mathbf{w}$ is the input and $\mathbf{u}^*, k^*$ is the solution of equation 14. This gives the coordinate-based representation $\mathbf{z}^* = \sigma^{k^*}(\mathbf{u}^*) = \Lambda(\mathbf{w})$. For smooth parameterizations $\sigma(\mathbf{u})$, the optimal solutions

$\mathbf{u}^*(\mathbf{w})$ satisfies the following implicit equation arising from the optimization problem,

$$G(\mathbf{u}^*, \mathbf{w}) := \nabla_{\mathbf{u}} \Phi_{k^*}(\mathbf{u}^*, \mathbf{w}) = 0. \tag{15}$$

During learning with back-propagation, we need to be able to compute the contributions to the loss function of the gradient

$$\nabla_\phi \mathbf{z}^* = \nabla_\phi \sigma^k(\mathbf{u}^*) = \nabla_\phi \Lambda \left(\tilde{\mathcal{E}}_\phi(\mathbf{x}) \right) = \nabla_{\mathbf{w}} \Lambda(\mathbf{w}) \nabla_\phi \tilde{\mathcal{E}}_\phi, \tag{16}$$

where $\mathbf{w} = \tilde{\mathcal{E}}_\phi$. If we approach training models using directly the expressions of equation 16, we would need ways to compute both the gradients $\nabla_\phi \tilde{\mathcal{E}}_\phi$ and $\nabla_{\mathbf{w}} \Lambda(\mathbf{w})$. While the gradients $\nabla_\phi \tilde{\mathcal{E}}_\phi$ can be obtained readily for many model classes, such as neural networks when using back-propagation, the gradients $\nabla_{\mathbf{w}} \Lambda(\mathbf{w})$ pose additional challenges. If Λ can be expressed analytically, then back-propagation techniques can be employed directly. However, in practice, Λ will often result from a numerical solution of the optimization problem in equation 13 which uses equation 14. We show how in this setting alternative approaches can be used to obtain the contributions to the loss function of the gradient $\nabla_\phi \mathbf{z}^* = \nabla_\phi \Lambda \left(\tilde{\mathcal{E}}_\phi(\mathbf{x}) \right)$.

To account for the contributions of such terms to the loss, we instead derive expressions for the variations of $\mathbf{u}^*$ and $\mathbf{w}$ using that they are related by $G(u^*, \mathbf{w}) = 0$. If we parameterize $\phi = \phi(\gamma)$ by some scalar parameter γ, then we have $G(\mathbf{u}^*(\gamma), \mathbf{w}(\gamma)) = 0$ for each γ. For example, this is motivated by taking $\mathbf{w}(\gamma) = \tilde{\mathcal{E}}_\phi(\mathbf{x}(\gamma))$ and $\phi = \phi(\gamma)$ for some path $(\mathbf{x}(\gamma), \phi(\gamma))$ in the input and parameter space $(\mathbf{x}, \phi) \in \mathcal{X} \times \mathcal{P}$. We can obtain the needed gradients by determining the variations of $\mathbf{u}^* = \mathbf{u}^*(\gamma)$ by using equation 15. This follows since $\mathbf{z}^* = \sigma^k(\mathbf{u}^*)$ and $\nabla_\phi \mathbf{z}^* = \nabla_{\mathbf{u}} \sigma^k(\mathbf{u}^*) \nabla_\phi \mathbf{u}^*$. The $\nabla_{\mathbf{u}} \sigma^k(\mathbf{u}^*)$ often can be readily obtained analytically or numerically from back-propagation. The more challenging term to compute is $\nabla_\phi \mathbf{u}^*$. This can be obtained from $d\mathbf{u}^*/d\gamma$ by considering a sufficiently rich variety of paths in parameter space $\phi = \phi(\gamma)$ that probe each direction. From equation 15, this allows us to express the gradients using the Implicit Function Theorem as

$$0 = \frac{d}{d\gamma} G(\mathbf{u}^*(\gamma), \mathbf{w}(\gamma)) = \nabla_{\mathbf{u}} G \frac{d\mathbf{u}^*}{d\gamma} + \nabla_{\mathbf{w}} G \frac{d\mathbf{w}}{d\gamma}. \tag{17}$$

The term typically posing the most significant computational challenge is $d\mathbf{u}^*/d\gamma$ since $\mathbf{u}^*$ is obtained numerically from the optimization problem in equation 14. We solve for it using the expression in equation 17 to obtain

$$\frac{d\mathbf{u}^*}{d\gamma} = -\left[\nabla_{\mathbf{u}} G \right]^{-1} \nabla_{\mathbf{w}} G \frac{d\mathbf{w}}{d\gamma}. \tag{18}$$

This only requires that we can evaluate for a given $(\mathbf{u}, \mathbf{w})$ the local gradients $\nabla_{\mathbf{u}} G$, $\nabla_{\mathbf{w}} G$, $d\mathbf{w}/d\gamma$, and that $\nabla_{\mathbf{u}} G$ is invertible. Computationally, this only requires us to find numerically the solution $\mathbf{u}^*$ and evaluate numerically the expression in equation 18 for a given $(\mathbf{u}^*, \mathbf{w})$. This allows us to avoid needing to compute directly $\nabla_{\mathbf{w}} \Lambda_{\mathbf{w}}(\mathbf{w})$. Now since $\nabla_\phi \mathbf{z}^* = \nabla_{\mathbf{u}} \sigma^k(\mathbf{u}^*) \nabla_\phi \mathbf{u}^*$ and since we can compute $\nabla_\phi \mathbf{u}^*$ using equation 18, we can numerically evaluate $\nabla_\phi \mathbf{z}^*$. This provides the needed gradients for training our models with manifold latent spaces.

For learning via back-propagation, we use these results to assemble the needed gradients. For our manifold encoder maps $\mathcal{E}_\theta = \Lambda(\tilde{\mathcal{E}}_\theta(\mathbf{x}))$, we use the following approach. Using $\mathbf{w} = \tilde{\mathcal{E}}_\theta(\mathbf{x})$, we first find numerically the closest point in the manifold $\mathbf{z}^* \in \mathcal{M}$ and represent it as $\mathbf{z}^* = \sigma(\mathbf{u}^*) = \sigma^{k^*}(\mathbf{u}^*)$ for some chart k^*. Next, using this chart we compute the gradients using that

$$G = \nabla_{\mathbf{u}} \Phi(\mathbf{u}, \mathbf{w}) = -(\mathbf{w} - \sigma(\mathbf{u}))^T \nabla_{\mathbf{u}} \sigma(\mathbf{u}). \tag{19}$$

We take in equation 19 a column vector convention with $\nabla_{\mathbf{u}} \sigma(\mathbf{u}) = [\sigma_{u_1} | \dots | \sigma_{u_k}]$. We next compute

$$\nabla_{\mathbf{u}} G = \nabla_{\mathbf{uu}} \Phi = \nabla_{\mathbf{u}} \sigma^T \nabla_{\mathbf{u}} \sigma - (\mathbf{w} - \sigma(\mathbf{u}))^T \nabla_{\mathbf{uu}} \sigma(\mathbf{u}) \tag{20}$$

8

and

$$\nabla_{\mathbf{w}} G = \nabla_{\mathbf{w},\mathbf{u}} \Phi = -I \nabla_{\mathbf{u}} \sigma(\mathbf{u}). \tag{21}$$

From equations 20 and 21, the gradients $\nabla_{\mathbf{u}} G$, $\nabla_{\mathbf{w}} G$, and using equation 18, we compute $\nabla_{\phi} \mathbf{z}^*$. This allows us to learn VAEs with latent spaces for $\mathbf{z}$ with general specified topologies and controllable geometric structures. We provide additional details in Appendix A. We refer to this learning framework for data-driven modeling of dynamics on latent spaces having general geometries as Geometric Dynamic Variational Autoencoders (GD-VAEs).

We remark that the geometry of manifolds can be represented in many different ways using either extrinsic or intrinsic descriptions. The algorithms leverage extrinsic descriptions which allows for avoiding the need for local coordinate charts and instead uses a global parameterization (over-parameterization) of the manifold. While many of the numerical calculations are performed in $\mathbb{R}^n$, we should emphasize that the learning methods still gain the benefits of the lower dimensionality of the manifold and still gain the benefits of the constraints arising from the latent-spaces geometric and topological structure. For instance, the decoder only needs to learn correct mappings from the subset of points that lie within the manifold. The manifold latent-space results in learning methods gaining statistical power from the training samples since their mappings to and from the manifold results overall in needing to learn responses over a smaller set of inputs.

The manifold latent spaces provided by GD-VAEs also can be combined with the use of tailored prior distributions $\mathfrak{p}(\mathbf{z})$ for the manifold setting. For simplicity, we primarily use here generic Gaussian priors similar to those used on the ambient space. When restricted to the geometric structure this gives a conditional probability distribution over the manifold. Our aim here is to use the priors to help with identifiability by setting the length-scale of the embedding maps and to help with learning by driving training steps toward a common target encoding. The GD-VAE methods are flexible and any prior for which the KL-Divergence regularization can be computed can be used in practice. Priors can be developed for specific manifold latent-spaces leveraging further knowledge and information relevant to a target machine learning task.

GD-VAEs provide ways to leverage topological and geometric information to help unburden the encoder and decoder from having to learn these embedding structures as part of training. As we shall discuss below in the context of specific examples, the manifold latent space structures also can help with reduced sensitivity to noise and obtaining more robust and parsimonious representations. This increased level of organizational structure also can be used to help with identifiability and with interpretability of the learned representations.

3. Results

We report on the performance of the methods. We consider examples that include learning the dynamics of the non-linear Burger's PDEs and reaction-diffusion PDEs. We also investigate learning representations for constrained mechanical systems. We consider the roles of manifold latent spaces having different dimensions, topology, and geometry.

3.1 Burgers' Equation of Fluid Mechanics: Learning Nonlinear PDE Dynamics

We consider the nonlinear viscous Burgers' equation

$$u_t = -uu_x + \nu u_{xx}, \tag{22}$$

where ν is the viscosity [6; 39]. We consider periodic boundary conditions on $\Omega = [0, 1]$. Burgers equation is motivated as a mechanistic model for the fluid mechanics of advective transport and shocks, and serves as a widely used benchmark for analysis and computational methods.

The nonlinear Cole-Hopf Transform $\mathcal{CH}$ can be used to relate Burgers equation to the linear Diffusion equation $\phi_t = \nu\phi_{xx}$ [39]. This provides a representation of the solution u

$$\phi(x,t) \;=\; \mathcal{CH}[u] = \exp\left(-\frac{1}{2\nu}\int_0^x u(x',t)dx'\right), \quad u(x,t) = \mathcal{CH}^{-1}[\phi] = -2\nu\frac{\partial}{\partial x}\ln\phi(x,t). \tag{23}$$

This can be represented by the Fourier expansion

$$\phi(x,t) = \sum_{k=-\infty}^{\infty} \hat{\phi}_k(0)\exp(-4\pi^2 k^2\nu t)\cdot\exp(i2\pi kx).$$

The $\hat{\phi}_k(0) = \mathcal{F}_k[\phi(x,0)]$ and $\phi(x,t) = \mathcal{F}^{-1}[\{\hat{\phi}_k(0)\exp(-4\pi^2 k^2\nu t)\}]$ with $\mathcal{F}$ the Fourier transform. This provides an analytic representation of the solution of the viscous Burgers equation $u(x,t) = \mathcal{CH}^{-1}[\phi(x,t)]$ where $\hat{\phi}(0) = \mathcal{F}[\mathcal{CH}[u(x,0)]]$. In general, for nonlinear PDEs with initial conditions within a class of functions $\mathcal{U}$, we aim to learn models that provide predictions $u(t+\tau) = \mathcal{S}_\tau u(t)$ approximating the evolution operator $\mathcal{S}_\tau$ over time-scale τ. For the Burgers equation, the $\mathcal{CH}$ provides in principle an analytic way to obtain a reduced order model by truncating the Fourier expansion to $|k| \le n_f/2$. This provides for the Burgers equation a benchmark model against which to compare our learned models. For general PDEs comparable analytic representations are not usually available. In addition, for many problems arising in practice, we are interested primarily in how the system behaves for a limited class of initial conditions and parameters. The aim then becomes to find a reduced model that predicts well the outcomes for these circumstances. We show how data-driven approaches can be developed for this purpose.

We develop VAE methods for learning reduced order models for the responses of nonlinear Burgers Equation when the initial conditions are from a collection of functions $\mathcal{U}$. We consider in particular $\mathcal{U} = \mathcal{U}_1 = \{u \,|\, u(x,t;\alpha) = \alpha\sin(2\pi x)+(1-\alpha)\cos^3(2\pi x)\}$. We remark that while the initial conditions are a linear combination of two functions, this relation does not hold for the solutions given the non-linear dynamics of the Burgers' PDE. We learn VAE models that extract from $u(x,t)$ latent variables $z(t)$ to predict $u(x,t+\tau)$. Given the non-uniqueness of representations and to promote interpretability of the model, we introduce the inductive bias that the evolution dynamics in latent space for z is linear of the form $\dot{z} = -\lambda_0 z$, giving exponential decay rate λ_0. For discrete times, we take $z_{n+1} = f_{\theta_\ell}(z_n) = \exp(-\lambda_0\tau)\cdot z_n$, where $\theta_\ell = (\lambda_0)$. We treated λ_0 as a hyperparameter in our studies with $\exp(-\lambda_0\tau) = 0.75$, but this could also be learned if additional criteria is introduced for the latent-space representations. The exponential decay is used to influence how the dynamical information is represented in latent space and also helps ensure dynamical stability for predictions. For the encoders and decoders, we consider general nonlinear mappings which are represented by deep neural networks.

We train the model on the pairs $(u(x,t), u(x,t+\tau))$ by drawing m samples of $u^{k_i}(x,t_i) \in \mathcal{S}_{t_i}\mathcal{U}$ which generates the evolved state under Burgers equation $u^{k_i}(x,t_i+\tau)$ over time-scale τ. We perform studies with parameters $\nu = 2\times 10^{-2}$, $\tau = 2.5\times 10^{-1}$ with our Deep Neural Networks (DNNs) with layer sizes (in)-400-400-(out), ReLU activations, and $\gamma = 0.5$, $\beta = 1$. The covariances $\mathbf{\Sigma}_e$ and $\mathbf{\Sigma}_d$ used are diagonal with initial standard deviations $\sigma_d = \sigma_e = 4\times 10^{-3}$ so that $\mathbf{\Sigma}_e = \mathrm{diag}(\sigma_e^2)$ and $\mathbf{\Sigma}_d = \mathrm{diag}(\sigma_d^2)$. We show results of our model predictions in Figure 4 and Table 1.

We investigate the importance of the non-linear approximation properties of our methods in capturing system behaviors. We do this by making comparisons with linear methods that include Dynamic Mode Decomposition (DMD) [74; 81], Proper Orthogonal Decomposition (POD) [15], and a linear variant of our GD-VAE approach. Recent CNN-AEs have also studied related advantages of non-linear approximations [51]. Some distinctions in our work is the use of VAEs to further regularize the autoencoders and using topological latent spaces to facilitate further capturing inherent structures. The DMD and POD are widely used and successful approaches that aim to find an optimal linear space on which to project the dynamics and learn a linear evolution law for system
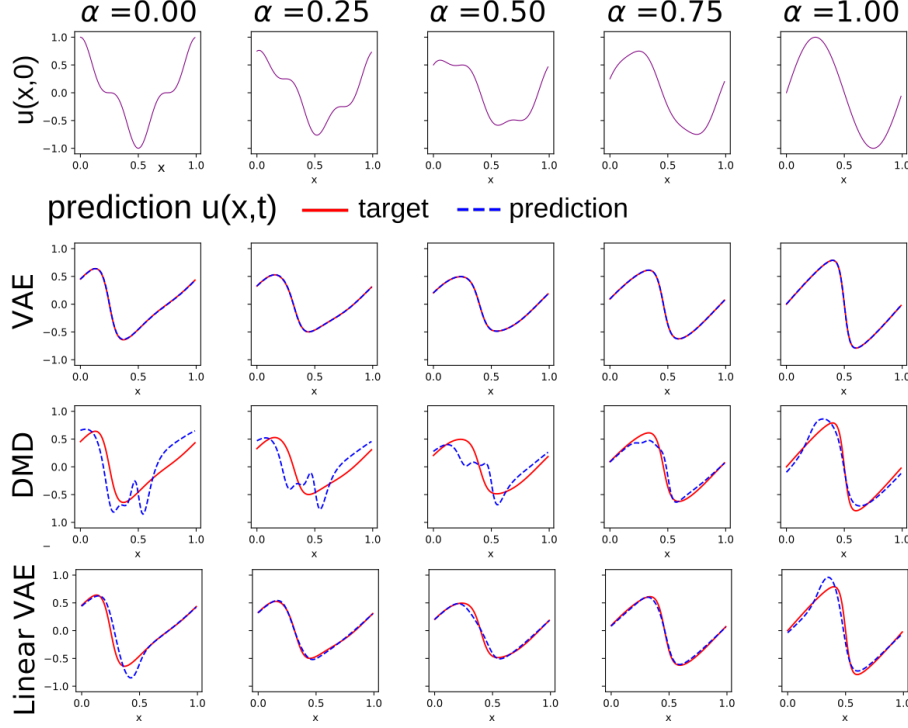
Figure 4: **Burgers' Equation: Prediction of Dynamics.** We consider responses for initial conditions $\mathcal{U}_1 = \{u \,|\, u(x,t;\alpha) = \alpha \sin(2\pi x) + (1 - \alpha) \cos^3(2\pi x)\}$. Predictions are made for the evolution u over the time-scale τ satisfying equation 22 with initial conditions in $\mathcal{U}_1$. We find our nonlinear VAE methods are able to learn with 2 latent dimensions the dynamics with errors $< 1\%$. Methods such as DMD [74; 81] with 3 modes which are only able to use a single linear space to approximate the initial conditions and prediction encounter challenges in approximating the nonlinear evolution. We find our linear VAE method with 2 modes provides some improvements by allowing for using different linear spaces for representing the input and output functions. Results are summarized in Table 1.

behaviors. DMD and POD have been successful in obtaining models for many applications, including steady-state fluid mechanics and transport problems [81; 74]. However, given their inherent linear approximations they can encounter well-known challenges related to translational and rotational invariances, as arise in advective phenomena and other settings [10]. Our comparison studies can be found in Table 1.

We also considered how our VAE methods performed when adjusting the parameter γ for the strength of the reconstruction regularization. The reconstruction regularization has a significant influence on how the VAE organizes representations in latent space and the accuracy of predictions of the dynamics, especially over multiple steps, see Figure 5 and Table 2. The regularization serves to align representations consistently in latent space facilitating multi-step compositions. We also found our VAE learned representations capable of some level of extrapolation beyond the training data, see Table 2. While extrapolation was not our main aim, it is interesting that the learned neural
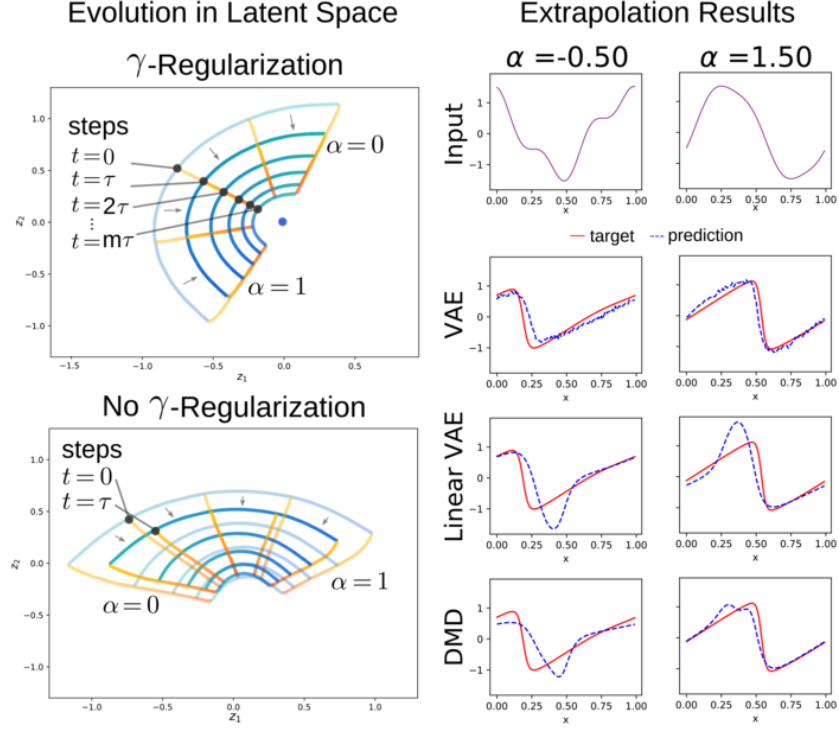
Figure 5: **Burgers' Equation: Latent Space Representations and Extrapolation Predictions.** We show the latent space representation z of the dynamics for the input functions $u(\cdot, t; \alpha) \in \mathcal{U}_1$. VAE organizes for u the learned representations $z(\alpha, t)$ in parameter α *(blue-green)* into circular arcs that are concentric in the time parameter t, *(yellow-orange)* *(left)*. The reconstruction regularization with γ aligns subsequent time-steps of the dynamics in latent space facilitating multi-step predictions. The learned VAE model exhibits a level of extrapolation to predict dynamics even for some inputs $u \notin \mathcal{U}_1$ beyond the training data *(right)*.

network representations appear to be based on features that retain enough information to be able to predict system responses reasonably even outside the training domain.

3.2 Burgers' Equation: Topological Structure of Parameterized PDEs

For parameterized PDEs and other systems, there is often significant prior geometric and topological information available concerning the space of responses of the system. For example, the parameters may vary periodically or have other imposed constraints limiting possible responses. We show how our methods can be utilized to make use of this structure to learn latent space representations having specified topological or geometric structures.

3.3 Periodic and Doubly-Periodic Systems: Cylinder and Torus Latent Spaces

We start by considering Burgers' equations with initial conditions parameterized periodically as

$$u_\alpha(x, t = 0) = \begin{bmatrix} \cos(2\pi\alpha) \\ \sin(2\pi\alpha) \end{bmatrix} \cdot \begin{bmatrix} \cos(2\pi x) \\ \sin(2\pi x) \end{bmatrix}, \qquad \alpha \in [0, 1]. \tag{24}$$

12

Method	Dim	0.25s	0.50s	0.75s	1.00s
GD-VAE	2	$4.27e\text{-}03 \pm 4.8e\text{-}04$	$4.49e\text{-}03 \pm 4.9e\text{-}04$	$4.76e\text{-}03 \pm 5.0e\text{-}04$	$6.43e\text{-}03 \pm 8.1e\text{-}04$
VAE 2D-Linear	2	$1.26e\text{-}01 \pm 2.7e\text{-}04$	$1.31e\text{-}01 \pm 1.6e\text{-}04$	$1.24e\text{-}01 \pm 2.1e\text{-}04$	$1.26e\text{-}01 \pm 2.4e\text{-}04$
DMD-3D	3	$2.21e\text{-}01 \pm 0.0e\text{-}00$	$1.79e\text{-}01 \pm 0.0e\text{-}00$	$1.56e\text{-}01 \pm 0.0e\text{-}00$	$1.49e\text{-}01 \pm 0.0e\text{-}00$
POD-3D	3	$3.24e\text{-}01 \pm 0.0e\text{-}00$	$4.28e\text{-}01 \pm 0.0e\text{-}00$	$4.87e\text{-}01 \pm 0.0e\text{-}00$	$5.41e\text{-}01 \pm 0.0e\text{-}00$
Cole-Hopf-2D	2	$5.18e\text{-}01 \pm 0.0e\text{-}00$	$4.17e\text{-}01 \pm 0.0e\text{-}00$	$3.40e\text{-}01 \pm 0.0e\text{-}00$	$1.33e\text{-}01 \pm 0.0e\text{-}00$
Cole-Hopf-4D	4	$5.78e\text{-}01 \pm 0.0e\text{-}00$	$6.33e\text{-}02 \pm 0.0e\text{-}00$	$9.14e\text{-}03 \pm 0.0e\text{-}00$	$1.58e\text{-}03 \pm 0.0e\text{-}00$
Cole-Hopf-6D	6	$1.48e\text{-}01 \pm 0.0e\text{-}00$	$2.55e\text{-}03 \pm 0.0e\text{-}00$	$9.25e\text{-}05 \pm 0.0e\text{-}00$	$7.47e\text{-}06 \pm 0.0e\text{-}00$

Table 1: **Burgers' Equation: Prediction Accuracy.** The reconstruction L^1-relative errors in predicting $u(x,t)$ and standard error over 5 training trials. We compare our VAE methods, Dynamic Model Decomposition (DMD), and Proper Orthogonal Decomposition (POD), and reduction by Cole-Hopf (CH), over multiple-steps and number of latent dimensions (Dim).

γ	0.00s	0.25s	0.50s	0.75s	1.00s
0.00	$1.20e\text{-}01 \pm 5.8e\text{-}03$	$3.77e\text{-}03 \pm 6.9e\text{-}04$	$9.38e\text{-}02 \pm 6.7e\text{-}03$	$1.84e\text{-}01 \pm 1.3e\text{-}02$	$2.83e\text{-}01 \pm 1.9e\text{-}02$
0.05	$4.18e\text{-}03 \pm 4.0e\text{-}04$	$3.80e\text{-}03 \pm 4.0e\text{-}04$	$4.00e\text{-}03 \pm 4.0e\text{-}04$	$4.24e\text{-}03 \pm 4.1e\text{-}04$	$4.71e\text{-}03 \pm 4.7e\text{-}04$
2.00	$5.20e\text{-}03 \pm 4.8e\text{-}04$	$5.03e\text{-}03 \pm 5.1e\text{-}04$	$5.30e\text{-}03 \pm 4.8e\text{-}04$	$5.55e\text{-}03 \pm 4.9e\text{-}04$	$8.52e\text{-}03 \pm 1.1e\text{-}03$

Table 2: **Burgers' Equation: Prediction Accuracy and Regularizations.** The reconstruction L^1-relative errors and standard error over 5 training trials. We show prediction of $u(x,t)$ varying the strength of the reconstruction regularization γ in equation 7.

We consider solutions with $x, t \in [0, 1]$. Since the boundary conditions are periodic $u(0, t) = u(1, t)$, the initial conditions parameterized by α effectively shift relative to one another and we have the topology of a circle. We also consider initial conditions parameterized doubly-periodic as

$$u_{\alpha_1, \alpha_2}(x, 0) = \begin{bmatrix} \cos(\alpha_1) \\ \sin(\alpha_1) \\ \cos(\alpha_2) \\ \sin(\alpha_2) \end{bmatrix} \cdot \begin{bmatrix} \cos(2\pi x) \\ \sin(2\pi x) \\ \cos(4\pi x) \\ \sin(4\pi x) \end{bmatrix}, \quad \alpha_1, \alpha_2 \in [0, 2\pi]. \tag{25}$$

This corresponds to the topology of a torus. We can project our solutions onto the Clifford Torus given by the product space $S^1 \times S^1 \subset \mathbb{R}^4$, where S^1 is the circle space [44; 83]. For example representations of latent spaces, see Figure 6. In each case, we seek representations that disentangle state information of the collection of solutions $\{u_\alpha\}_{0 \le \alpha \le 1}$ or $\{u_{\alpha_1, \alpha_2}\}_{0 \le \alpha_1, \alpha_2 \le 2\pi}$.

In principle, for the periodic case, one analytic way the family of solutions $\{u_\alpha\}_{0 \le \alpha \le 1}$ could be encoded continuously is to use the $\mathbb{R}^2$ latent space with mapping

$$u_\alpha(x, t) \to (t \cos(2\pi \alpha), t \sin(2\pi \alpha)). \tag{26}$$

This is similar to projecting a cone-shaped manifold in three dimensions to the two dimensional xy-plane. However, in practice, this is hard to learn from the training data given how this representation entangles the time and configuration state information, see Appendix B. Also, while such encodings may be useful for some problems, using models that disentangle time and the configuration state information into separate dimensions in the latent space is expected to result in simpler manifold structures and more interpretable results. We impose this structure by seeking representations of
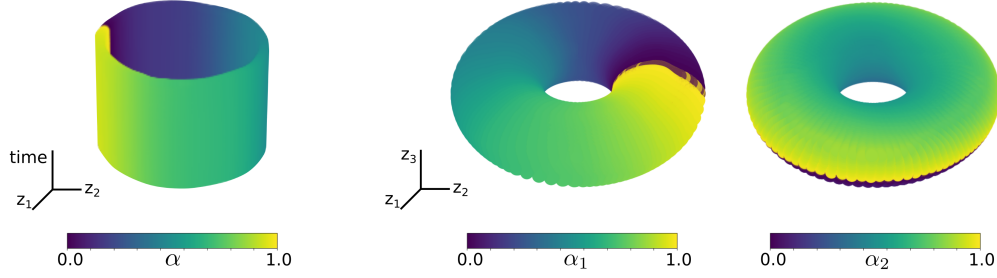
Figure 6: **GD-VAE Manifold Latent Spaces for Periodic and Doubly-Periodic Systems.**
We show results for the learned mappings of solutions of the parameterized Burgers Equation in the periodic α case and doubly-periodic α_1, α_2 case. For such time-dependent PDE solutions, it is natural to consider latent spaces with abstract cylinder-like shapes represented as a product-space $\mathcal{B} \times \mathbb{R}$. In this cylinder topology, the infinite axis $\mathbb{R}$ corresponds to time and the other dimensions to the state of the system for some base space $\mathcal{B}$. For a single periodic parameter, the base-space $\mathcal{B}$ is a circle giving the conventional cylinder shown on the *(left)*. For two doubly-periodic parameters, it is natural to consider a torus for the base-space $\mathcal{B}$. In practice, we find in training there are benefits to using the Clifford Torus $S^1 \times S^1 \subset \mathbb{R}^4$. For visualization purposes, we depict our results mapping this onto the conventional torus in $\mathbb{R}^3$ as shown on the *(right)*.

the form

$$u_{\boldsymbol{\alpha}}(x,t) \rightarrow (\mathbf{f}(\boldsymbol{\alpha}), t).$$

The mapping $\mathbf{f}$ does not depend on time and captures the current state information. As a consequence of the potentially non-trivial topology of the $\boldsymbol{\alpha}$ latent variable, Euclidean latent spaces with a continuous encoder that disentangles time must also map to a sufficiently larger number of dimensions to embed the manifold to preserve the topology. In the periodic case, this would be to at least three dimensions. An example of such an analytic mapping to $\mathbb{R}^3$ is

$$u_\alpha(x,t) \rightarrow (\cos(2\pi\alpha), \sin(2\pi\alpha), t).$$

This mapping corresponds in a latent space having a cylinder topology. Similarly, for the doubly-periodic family $\{u_{\alpha_1, \alpha_2}\}$ the state information can be mapped in principle to a torus by $\mathbf{f}(\boldsymbol{\alpha})$. This would yield the disentangled embedding $u_{\boldsymbol{\alpha}}(x,t) \rightarrow (\mathbf{f}(\boldsymbol{\alpha}), t)$ with $\boldsymbol{\alpha} = (\alpha_1, \alpha_2)$. The torus can be represented in principle in $\mathbb{R}^3$ or as a product space $S^1 \times S^1$ where S^1 is the circle space. This latter representation is referred to as the Clifford Torus [44; 83]. This gives the latent space with the cylinder topology $\mathcal{B} \times \mathbb{R}$, where $\mathcal{B}$ is the base-space. For periodic case the $\mathcal{B} = S^1$ is a circle and for the doubly-periodic case the $\mathcal{B} = S^1 \times S^1$ is a torus.

We demonstrate how our methods can be used in practice to learn representations for the dynamics of the Burgers' Equation solutions for latent spaces having specified topologies. We use a few approaches to leverage the prior geometric information about the system and to regulate how the model organizes the latent representations. This includes (i) specifying an explicit time-evolution map to disentangle time by requiring $f_{ev}(\mathbf{z}) = \mathbf{z} + \Delta t \mathbf{e}_n \in \mathbb{R}^n$, where Δt is the discrete time step and $\mathbf{e}_n = (0, \ldots, 0, 1)$ is the nth-standard basis vector. We also use (ii) projection of the representation to a manifold structure, here we project the first $n-1$ dimensions to impose the topology as discussed in Section 2. In this way, we can introduce inductive biases for general initial conditions leveraging topologic and geometric knowledge to regulate the latent representations toward having desirable properties.

For our example of the Burger's Equations 22 with the periodic initial conditions in equation 24, we train using 10^4 solutions sampled uniformly over α and time t. The samples are corrupted with a Gaussian noise having standard deviation $\sigma = 0.02$. We train GD-VAEs using our framework discussed in Section 1. The neural networks have the architecture with number of layers for encoders (in)-400-400-(latent-space) and decoders (latent-space)-400-400-(out). The layers have a bias except for the last layer and ReLU activation functions [31; 50]. The numerical solutions of the Burgers equation 22 are sampled as $u(x_k, t) = u_k$ with x_k at $n = 100$ points giving (in)=(out)=n=100. We also perform trainings representing the encoder variance with a trainable deep neural network, and set decoder variance such that log likelihood can be viewed as a Mean Square Error (MSE) loss with weight 10^{-4}.

We predict the future evolution of the solutions of the Burger's Equation over multiple steps using equation 1. We investigate the L^1-accuracy of the learned GD-VAE predictions relative to the numerical solutions. We compare the GD-VAE with more conventional VAE methods that do not utilize topological information. We also make comparisons with standard Autoencoders (AEs) with and without utilizing topological information and our projection approaches in Section 2. We refer in the notation to our geometric projection as *g-projection*. We show our results in Table 3.

We find our GD-VAE methods are able to learn parsimoneous disentangled representations that achieve a comparable reconstruction accuracy to standard VAE and AE provided these latter methods use enough dimensions. Reconstruction accuracy alone is only one way to measure the quality of the latent-space representations. Topological considerations play a significant role. When using too few dimensions, we see the standard AE and VAE methods can struggle to learn a suitable representation, see Figure 7. As we discuss in more detail in Appendix B, this arises since the autoencoders involve continuous maps that are unable to accomodate the topology injectively. For periodic systems this can result in rapid looping back behaviors approximating a discontinuity which results in scattered latent space points, see Figure 14. For additional discussions of the role of latent topology and training behaviors see Appendix B.

For longer-time multi-step predictions, we find our geometric projections can help with stability of the predictions arising from composition of the learned maps. We find both GD-VAEs and AEs with *g-projection* have enhanced stability when compared to standard AEs for multi-step predictions. We find at time $t = 1.00s$, the AE+*g-projection* still is able to make accurate predictions, while the standard AE incurs significant errors, see Table 3. This arises from the geometric constraints reducing the number of dimensions and the "volume" of the subset of points in the latent space overwhich the encoder and decoders must learn correct mappings for responses. The geometric projection also serves to enhance the statistical power of the training data given more opportunities for common overlap of cases. In contrast, for multi-step predictions by general autoencoder maps new latent-space codes can arise during compositions that are far from those encountered in the training set resulting in unknown behaviors for the dynamic predictions. While in some cases there is comparable L^1-reconstruction errors to AEs with *g-projection*, the VAEs give better overall representations and more reliable training since the noise regularizations result in smoother organization of the latent codes with less local sensitivity in the encoder-decoder maps relative to the AEs. The geometric projections developed for GD-VAEs provide further benefits for stability by constraining the latent-space dynamics to be confined within a closed manifold which further enhances the multi-step predictions, see results for $t = 1.00s$ in Table 3.

Periodic: Cylinder Topology.						
Method	Dim	0.00s	0.25s	0.50s	0.75s	1.00s
GD-VAE	3	$2.12e\text{-}02 \pm 9.3e\text{-}05$	$2.15e\text{-}02 \pm 1.5e\text{-}03$	$2.57e\text{-}02 \pm 2.7e\text{-}03$	$3.14e\text{-}02 \pm 4.0e\text{-}03$	$4.72e\text{-}02 \pm 5.5e\text{-}03$
VAE-2D	2	$2.51e\text{-}02 \pm 1.6e\text{-}03$	$2.33e\text{-}02 \pm 1.9e\text{-}03$	$2.93e\text{-}02 \pm 2.3e\text{-}03$	$4.01e\text{-}02 \pm 2.9e\text{-}03$	$6.42e\text{-}02 \pm 4.5e\text{-}03$
VAE-3D	3	$2.32e\text{-}02 \pm 3.8e\text{-}03$	$2.38e\text{-}02 \pm 2.9e\text{-}03$	$2.98e\text{-}02 \pm 3.4e\text{-}03$	$3.78e\text{-}02 \pm 4.0e\text{-}03$	$5.67e\text{-}02 \pm 5.6e\text{-}03$
VAE-10D	10	$1.99e\text{-}02 \pm 5.9e\text{-}04$	$1.99e\text{-}02 \pm 8.8e\text{-}04$	$2.49e\text{-}02 \pm 1.2e\text{-}03$	$3.20e\text{-}02 \pm 1.6e\text{-}03$	$4.88e\text{-}02 \pm 2.8e\text{-}03$
AE (g-projection)	3	$1.45e\text{-}02 \pm 7.7e\text{-}04$	$1.47e\text{-}02 \pm 8.1e\text{-}04$	$1.57e\text{-}02 \pm 9.0e\text{-}04$	$1.70e\text{-}02 \pm 9.8e\text{-}04$	$1.98e\text{-}02 \pm 9.7e\text{-}04$
AE (no projection)	3	$1.39e\text{-}02 \pm 4.6e\text{-}04$	$1.39e\text{-}02 \pm 6.0e\text{-}04$	$8.55e\text{-}02 \pm 5.9e\text{-}03$	$1.99e\text{-}01 \pm 1.1e\text{-}02$	$3.18e\text{-}01 \pm 1.3e\text{-}02$

Doubly-Periodic: Torus Topology.						
Method	Dim	0.00s	0.25s	0.50s	0.75s	1.00s
GD-VAE	5	$5.48e\text{-}02 \pm 3.8e\text{-}03$	$4.75e\text{-}02 \pm 4.4e\text{-}03$	$7.32e\text{-}02 \pm 5.8e\text{-}03$	$1.06e\text{-}01 \pm 7.3e\text{-}03$	$1.55e\text{-}01 \pm 1.0e\text{-}02$
VAE-3D	3	$2.37e\text{-}01 \pm 2.9e\text{-}03$	$2.22e\text{-}01 \pm 3.3e\text{-}03$	$4.09e\text{-}01 \pm 1.5e\text{-}02$	$6.39e\text{-}01 \pm 2.1e\text{-}02$	$7.97e\text{-}01 \pm 2.6e\text{-}02$
VAE-5D	5	$5.56e\text{-}02 \pm 1.3e\text{-}03$	$4.52e\text{-}02 \pm 1.8e\text{-}03$	$6.73e\text{-}02 \pm 2.7e\text{-}03$	$9.66e\text{-}02 \pm 4.0e\text{-}03$	$1.42e\text{-}01 \pm 5.7e\text{-}03$
VAE-10D	10	$5.45e\text{-}02 \pm 2.1e\text{-}03$	$4.61e\text{-}02 \pm 2.0e\text{-}03$	$7.12e\text{-}02 \pm 2.2e\text{-}03$	$1.01e\text{-}01 \pm 2.6e\text{-}03$	$1.47e\text{-}01 \pm 4.6e\text{-}03$
AE (g-projection)	5	$5.42e\text{-}02 \pm 6.4e\text{-}03$	$5.02e\text{-}02 \pm 5.9e\text{-}03$	$6.35e\text{-}02 \pm 9.2e\text{-}03$	$8.12e\text{-}02 \pm 1.3e\text{-}02$	$1.03e\text{-}01 \pm 1.8e\text{-}02$
AE (no projection)	5	$2.80e\text{-}02 \pm 5.4e\text{-}04$	$3.16e\text{-}02 \pm 3.9e\text{-}04$	$1.84e\text{-}01 \pm 6.9e\text{-}03$	$4.59e\text{-}01 \pm 3.1e\text{-}02$	$8.04e\text{-}01 \pm 6.3e\text{-}02$

Table 3: **Burgers' Equations for Periodic and Doubly-Periodic Parameterized PDEs: L^1-Prediction Accuracy.** The L^1-relative errors and standard error over 5 training trials. We train using the GD-VAEs framework discussed in Sections 1 and 2. We make comparisons to more conventional Variational Autoencoders (VAEs) and standard Autoencoders (AEs). The AE-(g-projection) method refers to the case where we combine using a standard AE (non-variational) and project the latent codes using our geometric projection approach developed in Section 2.
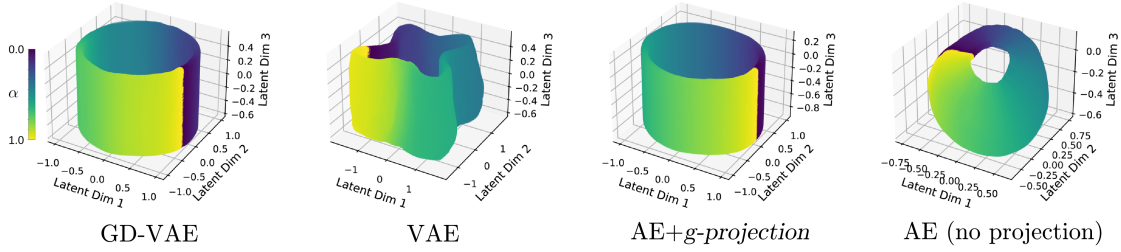

GD-VAE   VAE   AE+*g-projection*   AE (no projection)

Figure 7: **Learned Latent Spaces for the Burgers' Equations.** The methods learn different three dimensional latent space representations for the non-linear dynamics. The GD-VAE method uses a model that encodes state information of the solutions on a circle which is aligned temporally *(left)*. This allows for a representation of the evolution dynamics in the latent space following a simple contour in the direction of the Latent Dimension 3. While VAE without using a geometric prior learns an embedding having cylindrical topology the latent representation is geometrically distorted *(left-middle)*. The AE+*g-projection* is found to learn a latent space representation with disentangled time and state information and with encodings that are aligned temporally *(right-middle)*. The AE without using a geometric prior learns an arbitrary embedding that entangles the temporal and state information *(right)*. The general AE representations also do not have a preferred scale nor an alignment with the coordinate origin relative to the GD-VAE methods. The *g-projection* approach is discussed in more detail in Section 2.

3.4 Constrained Mechanical Systems: Learning with General Manifold Latent Spaces
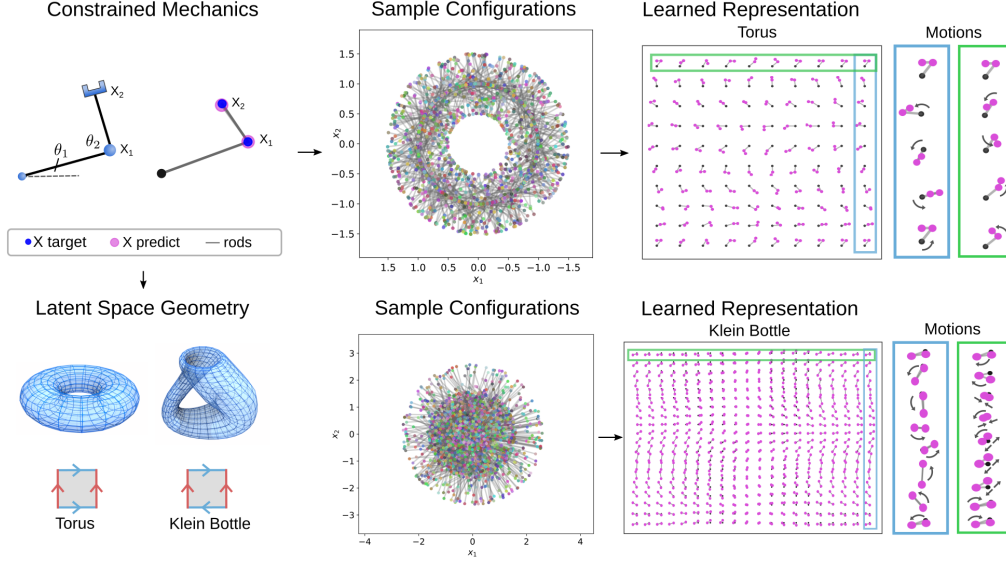

Figure 8: **GD-VAE Representations of Motions using Manifold Latent Spaces.** We learn from observations representations for constrained mechanical systems using manifold latent spaces of general shape $\mathcal{M}$. The arm mechanism has configurations $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2) \in \mathbb{R}^4$ *(left)*. For rigid segments, the motions are constrained to be on a manifold (torus) $\mathcal{M} \subset \mathbb{R}^4$. For extendable segments, we can also consider more exotic constraints, such as requiring $\mathbf{x}_1, \mathbf{x}_2$ to be on a klein bottle in $\mathbb{R}^4$ *(middle)*. Results of our GD-VAE methods for learned representations for motions under these constraints are shown on the *(right)*. GD-VAE learns the segment length constraint and two nearly decoupled coordinates for the torus data that mimics the roles of angles *(top-right)*. For the klein bottle case, GD-VAEs learns two rotation-like motions to generate the constrained configurations *(bottom-right)*.

We consider physical systems with constrained mechanics, such as the arm mechanism for reaching for objects in Figure 8. The observations are taken to be the two locations $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^2$ giving $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2) \in \mathbb{R}^4$. When the segments are constrained, these configurations lie on a manifold embedded in $\mathbb{R}^4$. The aim of these studies is to show how GD-VAEs with manifold latent spaces can be used to learn representations for constrained systems. We do not currently consider dynamical predictions in these studies. This corresponds to GD-VAEs with latent space dynamics $\mathbf{z}' = f_{\theta_\ell}(\mathbf{z}) = \mathbf{z}$ given by the identity map. We focus instead on the types of representations learned for the constrained latent geometries. When the two segments are constrained to be rigid, this results in the collection of configurations having the topology of a torus. We can also allow the segments to extend and consider more exotic constraints. For example, we could require the two points $\mathbf{x}_1, \mathbf{x}_2$ always be within a constraint set that is a klein bottle surface within $\mathbb{R}^4$. Related situations arise in other areas of imaging and mechanics, such as in pose estimation and in studies of visual perception [64; 13; 72].

For the arm mechanics, we can use this prior knowledge to construct a latent space having the topology of a torus represented by the product space of two circles $S^1 \times S^1$. To obtain a learnable class of manifold encoders, we use the family of maps $\mathcal{E}_\theta = \Lambda(\tilde{\mathcal{E}}_\theta(x))$. We take $\tilde{\mathcal{E}}_\theta(x)$ to map into $\mathbb{R}^4$ and $\Lambda(\mathbf{w}) = \Lambda(w_1, w_2, w_3, w_4) = (z_1, z_2, z_3, z_4) = \mathbf{z}$, where $(z_1, z_2) = (w_1, w_2)/\|(w_1, w_2)\|$ and $(z_3, z_4) = (w_3, w_4)/\|(w_3, w_4)\|$. This provides an analytic formulation we can use to perform gradient-based learning in combination with our back-propagation training methods in Section 2. For the

case of klein bottle constraints, we use a numerical approach based on a point-cloud representation of the non-orientable manifold with parameterized embedding in $\mathbb{R}^4$ given by

$$z_1 = (a + b\cos(u_2))\cos(u_1) \quad z_2 = (a + b\cos(u_2))\sin(u_1)$$
$$z_3 = b\sin(u_2)\cos\left(\frac{u_1}{2}\right) \qquad z_4 = b\sin(u_2)\sin\left(\frac{u_1}{2}\right).$$

We take $u_1, u_2 \in [0, 2\pi]$. The $\Lambda(\mathbf{w})$ is taken to be the map to the nearest point of the manifold $\mathcal{M}$, which we use to compute numerically the surface information. This allows for gradient-based learning when combined with our back-propagation training methods discussed in Section 2.

We train our GD-VAE methods with encoder and decoder DNNs having layers of sizes (in)-100-500-100-(out) with Leaky-ReLU activations with $s = 1e\text{-}6$ with results reported in Figure 8. The results demonstrate how we can learn smooth representations for constrained mechanical systems for both orientable and non-orientable manifold latent spaces. For both the torus and klein bottle latent spaces, we see smooth representations are learned for generating the configurations. For the latent torus manifold, the representation is comparable to the two angular degrees of freedom. This shows our GD-VAE approaches can be used as unsupervised learning methods for contrained mechanical systems to learn representations in manifold latent spaces with general topology and orientability.

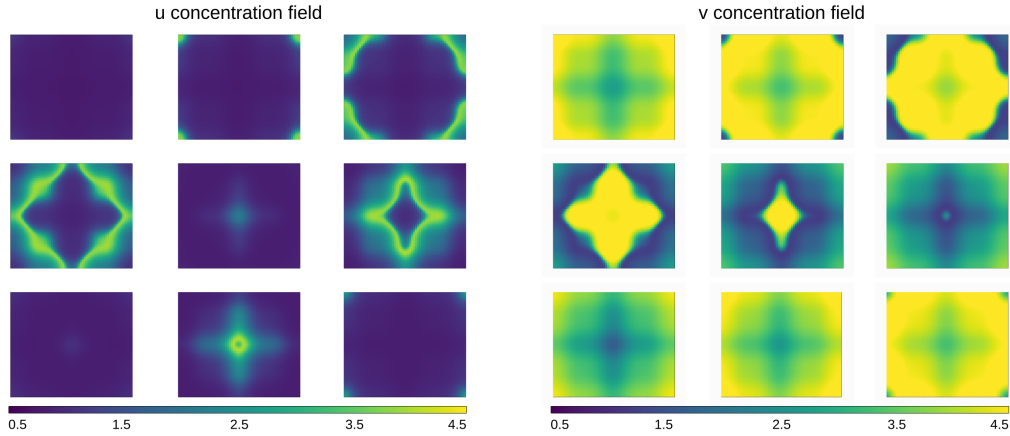### 3.5 Reaction-Diffusion PDEs in 2D: Learning Representations for Spatially Distributed Dynamics

Figure 9: **Brusselator Dynamics: Concentration Fields u and v.** We show the dynamics of the concentration fields $u(x,t)$ and $v(x,t)$ of the reaction-diffusion PDEs in equation 27 with initial conditions for $\alpha = 0.0$ in equation 28. Shown is the evolution of the concentration fields u, v from left to right starting at time $t = 60\Delta t$ to time $t = 140\Delta t$. After a transient, the concentration field dynamics approach a limit cycle having approximately periodic dynamics.

We show how our GD-VAEs can be used to learn features for representing the states and dynamic evolution of spatially extended reaction-diffusion systems, see Figure 9. Consider the system

$$\frac{\partial u}{\partial t} = D_1 \Delta u + f(u, v), \qquad \frac{\partial v}{\partial t} = D_2 \Delta v + g(u, v). \qquad (27)$$

The $u = u(x, t)$ and $v(x, t)$ give the spatially distributed concentration of each chemical species at time t with $x \in \mathbb{R}^2$. We consider the case with periodic boundary conditions with $x \in [0, L] \times [0, L]$.

We develop learning methods for investigating the Brusselator system [67; 66], which is known to have regimes exhibiting limit cycles [77; 35; 36]. This indicates after an initial transient, the orbit of the dynamics will localize and approach a subset of states topologically similar to a circle. We show how GD-VAE can utilize this topological information to construct latent spaces for encoding states of the system. The Brusselator [67; 66] has reactions with $f(u,v) = a - (1 + b)u + vu^2$ and $g(u,v) = bu - vu^2$. We take throughout the diffusivity $D_1 = 1, D_2 = 0.1$ and reaction rates $a = 1, b = 3$.

We consider initial conditions for the concentration fields u and v parameterized by $\alpha \in [0, 1]$ given by

$$u(x,y) = \alpha \sin(2\pi x/L_x) + (1 - \alpha)\cos^3(2\pi x/L_x) \tag{28}$$
$$v(x,y) = \alpha \sin(2\pi y/L_y) + (1 - \alpha)\cos^3(2\pi y/L_y). \tag{29}$$

We remark that while the initial conditions are given by a linear combination of functions the resulting solutions do not satisfy this relation given the non-linear dynamics of the reaction-diffusion system. The chemical fields evolve under periodic boundary conditions within a box length $L_x = L_y = 64$. We use second order Central-Differences to estimate the spatial derivatives and Backward-Euler method for the temporal evolution [40; 12] (py-pde python package [87] is used for numerical calculations). For the discretization, we space the grid points with $\Delta x = 1.0$ and use a time-step of $\Delta t = 10^{-3}$.
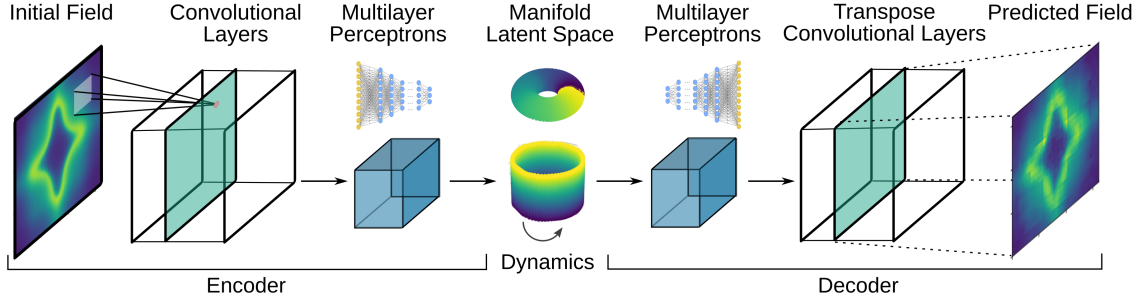

Figure 10: **GD-VAE Convolutional Neural Network Architecture.** For spatially extended fields u, v, we use architectures based on Convolutional Neural Networks (CNNs) consisting of the following processing steps. For the encoder, we extract features from the spatial fields using Convolutional Neural Networks (CNNs) which are fed into dense Multilayer Perceptrons (MLPs). The encoder serves to map the spatial fields u, v to codes **z** in a manifold latent space having prescribed topology using our methods discussed in Section 2. For the decoder, we use MLPs that feed into Transpose Convolutional Neural Networks (T-CNNs) that serve to reconstruct from **z** the spatial fields u, v We learn the encoders and decoders using our GD-VAE training framework discussed in Section 1.

We show how our GD-VAE methods can be used for solutions to learn a parsimonious disentangled representations in a manifold latent space. The dynamics consists of a brief transient followed by close approach to a limit cycle. For the dynamics after the transient, we encode the temporal and state information using a cylindrical topology. While a preliminary application of Principle Component Analysis (PCA) in a sufficiently large number of dimensions can be used to identify the transient and the period of the limit cycle, it does not provide a well-organized or parsimonious representation of the dynamics. Using from PCA the top three singular vectors, we show the entangled embedding in Figure 11.
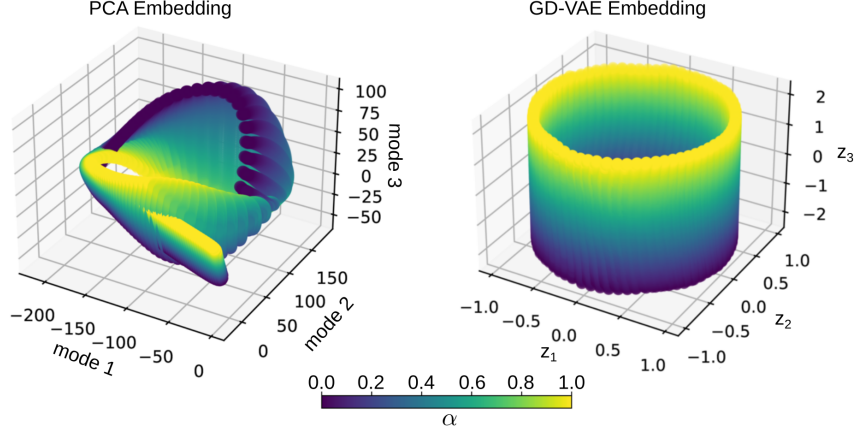
Figure 11: **Embeddings for Brusselator Reaction-Diffusion Dynamics.** A Principle Compo-
nent Analysis (PCA) embedding is compared to GD-VAE embeddings for solutions of
the PDEs in equation 27. While the PCA embedding provides some information on the
transient and limit cycle behaviors, the latent space representation entangles the tem-
poral and state information, *(left)*. The GD-VAE allows for learning a more organized
and parsemoneous embedding for the dynamics where state information is captured by
(z_1, z_2) and time by z_3, *(right)*.

We use GD-VAE methods to learn a more organized encoding for representing the states and
system dynamics. For this purpose, we develop encoders based on Convolutional Neural Networks
(CNNs) and Multilayer Perceptrons (MLPs) to map from spatial concentration fields u, v to latent
codes $\mathbf{z}$. The encoder serves to map the spatial fields u, v to codes $\mathbf{z}$ in a manifold latent space
having prescribed topology using our methods discussed in Section 2. We develop decoders based on
using MLPs that control Transpose Convolutional Neural Networks (T-CNNs) to map from latent
codes $\mathbf{z}$ to construct spatial concentration fields u, v, see Figure 10.

We specify a manifold latent space having the geometry of a cylinder $\mathcal{M} = \mathcal{B} \times \mathbb{R}$ with $\mathcal{B} = S^1$
and axis in the z_3-direction. We prescribe on this latent space the dynamics having the rotational
evolution

$$\mathbf{z}(t + \Delta t) = \begin{bmatrix} \cos(\omega \Delta t) & -\sin(\omega \Delta t) & 0 \\ \sin(\omega \Delta t) & \cos(\omega \Delta t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{z}(t). \tag{30}$$

This is expressed in terms of an embedding in $\mathbb{R}^3$. The ω gives the angular velocity. This serves to
regularize how the encoding of the reaction-diffusion system is organized in latent space.

The GD-VAE then is tasked with learning the encoding and decoding mappings for the reaction-
diffusion concentration fields u, v to representations that are well-organized for capturing and pre-
dicting the reaction-diffusion system states and dynamics. This organization allows for making
multi-step predictions from the latent space. Throughout our empirical studies, we use the angular
velocity $\omega = 0.282$ determined from the preliminary PCA analysis. In principle, such parameters
also can be learned as part of the training of the GD-VAE.

For training, we use an architecture with CNNs having four layers each of which has (in-channels,
out-channels, kernel-size, stride, padding) with parameters as follows respectively (2,10,3,3,1),
(10,20,3,3,1),(20,40,2,2,1),(40,100,5,1,0). Each layer was followed by a *ReLU* activation, except for
the last layer which is fed into the MLPs. The MLPs have an architecture with layer sizes 100-

(latent-space-embed) where latent-space-embed=3. The *g-projection* is then applied to the output of the MLPs to map to the manifold latent space $\mathcal{M}$ using our methods discussed in Section 2. For the decoder, we use MLPs with layer sizes (latent-space-embed)-100. We use T-CNNs having four layers each of which has (in-channels, out-channels, kernel-size, stride, padding) with parameters (100, 40, 5, 1, 0), (40, 20, 2, 2, 1), (20, 10, 3, 3, 1), (10, 2, 3, 3, 1). All layers have a bias and *ReLU* activations except for the last layer.

We use our GD-VAE methods for learning representations and for predicting the evolution dynamics of the reaction-diffusion system. We show a few predictions of the concentration fields by the GD-VAEs in comparison to the numerical solutions of the PDE in Figure 12. We characterize the accuracy of the predictions and reconstructions using L^1-relative errors. We report results of our methods for multi-step predictions in Table 4.
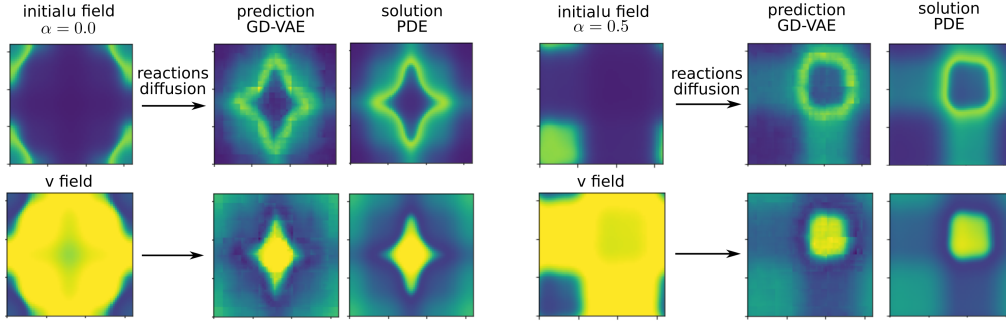

Figure 12: **Brusselator: GD-VAE Prediction of Concentration Fields u and v.** We show initial concentrations of two chemical species with spatial distribution given by the fields u and v *(left)*. We consider how these evolve after undergoing reaction and diffusion processes according to the PDEs in equation 27. We show the GD-VAE predictions for the spatial fields u, v with those of the numerical solutions of the PDEs *(right)*. We show the cases for $\alpha = 0.0$ and $\alpha = 0.5$ starting at $t = 15s$ and predicting the concentration fields u, v at time $t = 17s$. The L^1-relative errors of predictions are given in Table 4.

Method	Dim	0.00s	2.00s	4.00s	6.00s	8.00s
GD-VAE	3	$3.16e\text{-}02 \pm 5.4e\text{-}03$	$2.87e\text{-}02 \pm 3.8e\text{-}03$	$3.63e\text{-}02 \pm 7.0e\text{-}03$	$3.77e\text{-}02 \pm 6.8e\text{-}03$	$3.56e\text{-}02 \pm 8.3e\text{-}03$
VAE-3D	3	$2.61e\text{-}02 \pm 2.9e\text{-}03$	$2.36e\text{-}02 \pm 2.4e\text{-}03$	$2.08e\text{-}01 \pm 1.8e\text{-}01$	$2.61e\text{-}01 \pm 2.3e\text{-}01$	$2.16e\text{-}01 \pm 1.9e\text{-}01$
AE (g-projection)	3	$2.96e\text{-}02 \pm 3.5e\text{-}03$	$2.75e\text{-}02 \pm 2.8e\text{-}03$	$3.91e\text{-}02 \pm 9.7e\text{-}03$	$4.25e\text{-}02 \pm 1.2e\text{-}02$	$2.99e\text{-}02 \pm 2.5e\text{-}03$
AE (no projection)	3	$2.36e\text{-}02 \pm 1.8e\text{-}03$	$2.19e\text{-}02 \pm 1.5e\text{-}03$	$3.49e\text{-}01 \pm 3.2e\text{-}01$	$1.88e\text{-}01 \pm 1.6e\text{-}01$	$1.55e\text{-}01 \pm 1.3e\text{-}01$

Table 4: **Brusselator: L^1-Prediction Accuracy.** The reconstruction L^1-relative errors and standard error over 5 training trials. We compare the GD-VAE approach with other methods for learning latent space representations for the reaction-diffusion dynamics. We consider the relative error of predictions when receiving the concentration field at time t_1 and making a prediction of the concentration field at time $t_1 + \tau$. We investigate the multi-step predictions up to time-scale $\tau = 8.00s$. This involves a composition of the single-step predictions over multiple steps for dynamics that has gone entirely through at least one period of the limit cycle. We see the geometric structure in the latent space greatly enhances the stability and resulting accuracy of predictions.

We find that GD-VAEs are able to learn representations in the manifold latent spaces capable of making good predictions of the dynamics. The manifold latent space is particularly advantageous for multi-step predictions by helping to constrain the encodings promoting more robust learning over a lower dimensional space and smaller subset of points. In this geometric setting, learning only needs to be performed over the subset of points on the manifold. This enhances the statistical power of the training data. This also simplifies the type of encoder and decoder response functions that need to be learned relative to the higher dimensional setting of $\mathbb{R}^n$. In addition, during multi-step predictions we see the geometric constraints also enhance stability. This arises from the latent space dynamics always being confined within the manifold. As a result, it is less likely for the one-step updates to map to a code $\mathbf{z}$ in unfamiliar parts of the latent space away from those locations characterized during training. The lower dimension of the geometric latent space also provides similar benefits with encoding. This arises from points being more concentrated and providing more statistical power in learning a model for the local responses of the underlying reaction-diffusion system. The more organized representations are also more amenable to interpretation.

These results indicate some of the ways our GD-VAE approaches can be used to leverage qualitative information about the dynamics both to enhance learning and to gain insights into system dynamics. The methods provide practical ways to learn parsimonious representations capable of providing quantitatively accurate predictions for high dimensional dynamical systems having latent geometric structures. The GD-VAEs can be used to obtain representations for diverse learning tasks for many types of dynamical systems.

Software Package

We have developed software packages for the introduced GD-VAE methods. This includes python implementations of our geometric projections (g-projection), manifold latent space representations, custom back-propagation approaches for modular training, and examples. The python package can be installed using `pip install gd-vae-pytorch` or downloaded from our website. For more information, see http://atzberger.org/.

Conclusions

We introduced GD-VAEs for learning representations of nonlinear dynamics on manifold latent spaces having general topologies and geometries. The methods allow for learning representations with prescribed geometric properties which can be used to help enhance the robustness of dynamical predictions, yield more interpretable results, or provide further insights into the underlying mechanisms generating observed behaviors. The methods also allow for leveraging qualitative information and analysis of dynamical systems for use in data-driven learning. We performed several benchmark studies to validate and characterize the methods. This included making comparisons of GD-VAEs with POD, DMD, and more conventional AEs. Our results indicate how the non-linear approximation properties of neural networks combined with geometric inductive biases can be used to help improve the reductions of representations and the accuracy of predictions and reconstructions. We also presented results for constrained mechanical systems, and the non-linear dynamics of parameterized PDEs, Burgers' equations, and reaction-diffusion systems. The results indicate some of the ways geometric and topological information present opportunities to simplify model representations, aid in interpretability, and enhance robustness of predictions. The GD-VAEs can be used to obtain representations for diverse types of learning tasks involving dynamics.

Acknowledgements

Appendix

A. Backpropagation of Encoders for Non-Euclidean Latent Spaces given by General Manifolds

We develop methods for using back-propagation to learn encoder maps from $\mathbb{R}^d$ to general manifolds $\mathcal{M}$. We perform learning using the family of manifold encoder maps of the form $\mathcal{E}_\theta = \Lambda(\tilde{\mathcal{E}}_\theta(x))$. This allows for use of latent spaces having general topologies and geometries. We represent the manifold as an embedding $\mathcal{M} \subset \mathbb{R}^{2m}$ and computationally use point-cloud representations along with local gradient information, see Figure 13 and 3. To allow for $\mathcal{E}_\theta$ to be learnable, we develop approaches for incorporating our maps into general back-propagation frameworks.
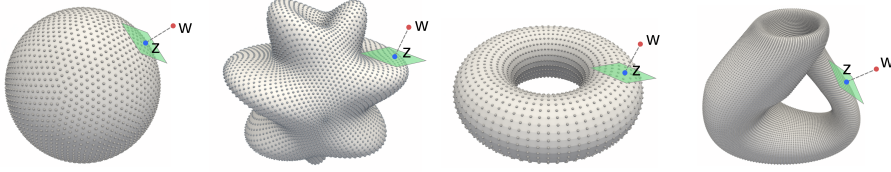

Figure 13: **Learnable Mappings to Manifold Surfaces.** We develop methods based on point cloud representations embedded in $\mathbb{R}^n$ for learning latent manifold representations having general geometries and topologies.

For a manifold $\mathcal{M}$ of dimension m, we can represent it by an embedding within $\mathbb{R}^{2m}$, as supported by the Whitney Embedding Theorem [85]. We let $\mathbf{z} = \Lambda(\mathbf{w})$ be a mapping $\mathbf{w} \in \mathbb{R}^{2m}$ to points on the manifold $\mathbf{z} \in \mathcal{M}$. This allows for learning within the family of manifold encoders $w = \tilde{\mathcal{E}}_\theta(x)$ any function from $\mathbb{R}^d$ to $\mathbb{R}^{2m}$. This facilitates use of deep neural networks and other function classes. In practice, we shall take $\mathbf{z} = \Lambda(\mathbf{w})$ to map to the nearest location on the manifold. We can express this as the optimization problem

$$z^* = \arg\min_{z \in \mathcal{M}} \frac{1}{2}\|w - z\|_2^2.$$

We can always express a smooth manifold using local coordinate charts $\sigma^k(u)$, for example, by using a local Monge-Gauge quadratic fit to the point cloud [32]. We can express $z^* = \sigma^k(u^*)$ for some chart k^*. In terms of the coordinate charts $\{\mathcal{U}_k\}$ and local parameterizations $\{\sigma^k(u)\}$ we can express this as

$$u^*, k^* = \arg\min_{k, u \in \mathcal{U}_k} \frac{1}{2}\|w - \sigma^k(u)\|_2^2,$$

where $\Phi_k(u, w) = \frac{1}{2}\|w - \sigma^k(u)\|_2^2$. The w is the input and u^*, k^* is the solution sought. For smooth parameterizations, the optimal solution satisfies

$$G = \nabla_z \Phi_{k^*}(u^*, w) = 0.$$

During learning we need gradients $\nabla_w \Lambda(w) = \nabla_w z$ when w is varied characterizing variations of points on the manifold $z = \Lambda(w)$. We derive these expressions by considering variations $w = w(\gamma)$ for a scalar parameter γ. We can obtain the needed gradients by determining the variations of $u^* = u^*(\gamma)$. We can express these gradients using the Implicit Function Theorem as

$$0 = \frac{d}{d\gamma}G(u^*(\gamma), w(\gamma)) = \nabla_u G \frac{du^*}{d\gamma} + \nabla_w G \frac{dw}{d\gamma}.$$

24

This implies

$$\frac{du^*}{d\gamma} = -\left[\nabla_u G\right]^{-1} \nabla_w G \frac{dw}{d\gamma}.$$

As long as we can evaluate at u these local gradients $\nabla_u G$, $\nabla_w G$, $dw/d\gamma$, we only need to determine computationally the solution u^*. For the back-propagation framework, we use these to assemble the needed gradients for our manifold encoder maps $\mathcal{E}_\theta = \Lambda(\tilde{\mathcal{E}}_\theta(x))$ as follows.

We first find numerically the closest point in the manifold $z^* \in \mathcal{M}$ and represent it as $z^* = \sigma(u^*) = \sigma^{k^*}(u^*)$ for some chart k^*. In this chart, the gradients can be expressed as

$$G = \nabla_u \Phi(u, w) = -(w - \sigma(u))^T \nabla_u \sigma(u).$$

We take here a column vector convention with $\nabla_u \sigma(u) = [\sigma_{u_1} | \dots | \sigma_{u_k}]$. We next compute

$$\nabla_u G = \nabla_{uu}\Phi = \nabla_u \sigma^T \nabla_u \sigma - (w - \sigma(u))^T \nabla_{uu}\sigma(u)$$

and

$$\nabla_w G = \nabla_{w,u}\Phi = -I\nabla_u \sigma(u).$$

For implementation it is useful to express this in more detail component-wise as

$$[G]_i = -\sum_k (w_k - \sigma_k(u))\partial_{u_i}\sigma_k(u),$$

with

$$
\begin{aligned}
{[\nabla_u G]}_{i,j} &= [\nabla_{uu}\Phi]_{i,j} = \sum_k \partial_{u_j}\sigma_k(u)\partial_{u_i}\sigma_k(u) \\
&\quad - \sum_k (w_k - \sigma_k(u))\partial^2_{u_i, u_j}\sigma_k(u) \\
{[\nabla_w G]}_{i,j} &= [\nabla_{w,u}\Phi]_{i,j} \\
&= -\sum_k \partial_{w_j} w_k \partial_{u_i}\sigma_k(u) = -\partial_{u_i}\sigma_j(u).
\end{aligned}
$$

The final gradient is given by

$$\frac{d\Lambda(w)}{d\gamma} = \frac{dz^*}{d\gamma} = \nabla_u \sigma \frac{du^*}{d\gamma} = -\nabla_u \sigma \left[\nabla_u G\right]^{-1} \nabla_w G \frac{dw}{d\gamma}.$$

In summary, once we determine the point $z^* = \Lambda(w)$ we need only evaluate the above expressions to obtain the needed gradient for learning via back-propagation

$$\nabla_\theta \mathcal{E}_\theta(x) = \nabla_w \Lambda(w) \nabla_\theta \tilde{\mathcal{E}}_\theta(x), \quad w = \tilde{\mathcal{E}}_\theta(x).$$

The $\nabla_w \Lambda$ is determined by $d\Lambda(w)/d\gamma$ using $\gamma = w_1, \dots w_n$. In practice, the $\tilde{\mathcal{E}}_\theta(x)$ is represented by a deep neural network from $\mathbb{R}^d$ to $\mathbb{R}^{2m}$. In this way, we can learn general encoder mappings $\mathcal{E}_\theta(x)$ from $x \in \mathbb{R}^d$ to general manifolds $\mathcal{M}$.

B. Role of Latent Space Geometry in Training

In practice, the encodings that organize the data in a manner convenient for representing the dynamic evolution can be challenging to learn in data-driven methods. In cases where the embedding space is too low dimensional or incompatible with the intrinsic topology this is further compounded. To illustrate some of the issues that arise, we consider a dynamical system with a periodic parameterization using as our example the Burgers' equation in equation 22 discussed in Section 3.1.

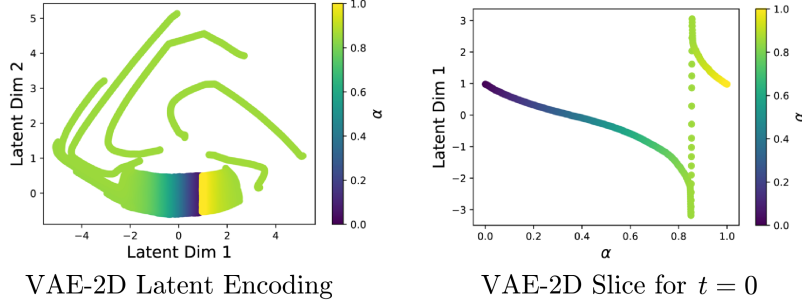VAE-2D Latent Encoding   VAE-2D Slice for $t = 0$

Figure 14: **Latent Space with Incompatible Geometry** We show a typical behavior encountered during training for dynamical systems having periodic parameterization $\alpha \in [0, 1]$ when using a latent space representation based on $\mathbb{R}^2$. While for a finite duration of time $t \in [0, T]$ there are in principle embeddings for such dynamics into $\mathbb{R}^2$ as in equation 26, these can be difficult to find during training, and further not possible to disentangle time and state configurations into separate dimensions. From the periodicity of inputs and the continuity of the encoder a typical behavior instead is for most states to be mapped to a patch and for the mapping to incur a penalty by rapidly loop back over only a small subset of inputs, here for responses for $\alpha \sim 0.85$. The continuous function approximates what would have been a discontinuity *(right)*. This rapid variation of the encoder results in states being mapped to a scattered set of codes which provide a poor representation for the dynamics. This is exhibited by the long spirals that show segments of the dynamics over time and how the states are mapped to the latent codes for $\alpha \sim 0.85$, *(left)*. Our GD-VAEs provide approaches to avoid this issue by allowing for general manifold latent spaces which maintain low dimension while accommodating such topology.

For this data set, it is natural to consider representing the system state and the dynamics using the geometry of a cylinder. In principle, it is possible to obtain a representation of the dynamics if restricted over a finite duration of time using only a two dimensional Euclidean space $\mathbb{R}^2$ as in equation 26. This is similar to a projection of a cone-shaped manifold in three dimensions projected to the two dimensional plane.

However, in practice this can be difficult to work with since there is an inherent tension during training between the periodicity and the natural ways the typical encoders will operate. This arises from the continuity of the encoder model class used in training that maps from the dynamical system solutions to the $\mathbb{R}^2$ latent space. We find that training proceeds initially by mapping a collection of states to a smooth patch of codes in the latent space. As a consequence of continuity, the encoder for the states associated with system responses for α and $\alpha' \equiv \alpha \pm 1.0 \mod 1.0$ need to map to similar codes. This results in an encoder that exhibits for a subset of states an extreme sensitivity to inputs that results in rapidly varying the code to loop back to accommodate the periodicity, see $\alpha = \lim \uparrow 0.85$ and $\alpha' = \lim \downarrow 0.85$ in Figure 14. Again, the issue arises since the states must map continuously to the latent space for all encoders encountered during training. Unless the map happens already to form a circle or other periodic structure in $\mathbb{R}^2$ there will be a subset of points that arise with rapid variation. This results in a subset of points having a poor encoding with states mapped to codes scattered in the latent space. This provides a poor basis for representing and predicting the system dynamics. When encoding over time we see disorganized fragments, connected again by transitions having rapid variations. We show this typical behavior observed during such training in Figure 14. This indicates the importance of choosing a latent space with

either a sufficiently large number of dimensions or which has a topology that is compatible with the data set. Our GD-VAE approach allows for keeping the latent space dimensionality small by allowing for accommodating general topologies in the latent space using our methods in Section 2.

C. Identifying Latent Geometric Structures using VAE Covariance Σ_e

The VAE training can be performed with a learnable covariance structure Σ_e with entries $[\Sigma_e]_{ij} = \Sigma_{e,ij}$, see equation 7. In VAE the KL-divergence regularization term is typically used to drive the encoding points to match a standard Gaussian distribution or other prior. The coding-point distribution arises from a combination of the scattering of the coding of points from the encoded training data points and from the Gaussian noise controlled by the variance Σ_e, see equation 7. Consider covariance $\Sigma_e = \text{diag}(\sigma_{e,i}^2)$, where $\sigma_{e,i}^2 = \Sigma_{e,ii}$. If components of $\sigma_{e,i}^2$ are held fixed to be a small value then the coding-point distribution can only arise from scattering of the encoding points. However, if $\sigma_{e,i}^2$ is learnable, then the encoder can map to a narrow distribution in some of the coding points with the Gaussian noise with variance $\sigma_{e,i}^2$ compensating to satisfy the KL-regularization term.
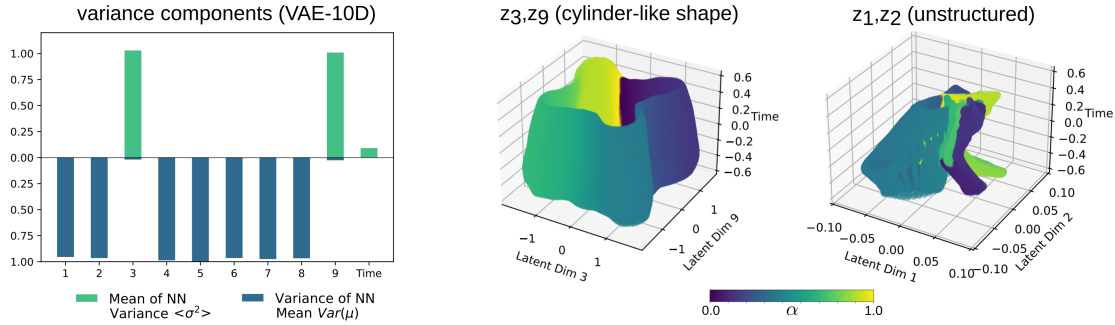

Figure 15: **Extracting Latent Geometric Structure using Learned VAE Variances.** We use the $\sigma_{e,i}$ and $\mu_{e,i}$ of VAEs to further extract geometric information from data and to perform reductions similar to [54]. By looking at mini-batches of data, we can compute two statistics (i) the mean of the variance (mov) $q_{i,mov}$ and (ii) the variance of the mean (vom) $q_{i,vom}$, see equations 31 and 32. We show results for Burgers' Equation with embedding in an unconstrained 10 dimensional space. We see the most relevant feature dimensions are the ones with $q_{i,vom}$ large and $q_{i,mov}$ small *(left)*. This yields a reduced three dimensional embedding with a good representation for the state information using z_1, z_9 coordinates *(middle)*. In contrast, using two arbitary dimensions such as coordinates z_1, z_2 yields a poor representation which is entangled and mapped to small scales *(right)*. This provides further ways to extract geometric information from the data and representations of latent structures.

We have found in practice in some cases this can be used to identify geometric structure within the data. We perform the embedding into a higher dimensional space than is needed, we find smaller values of Σ_e correlate with encoding to a lower dimensional manifold structure, similar to [54]. We show how this can be used in practice to find reduced embeddings. By looking at mini-batches of data with N_b samples, we can compute two statistics (i) the mean of the variance (mov)

$$q_{i,mov} = \frac{1}{N_b} \sum_{\ell=1}^{N_b} \sigma_{e,i}^2(\mathbf{x}^{(\ell)}), \tag{31}$$

and (ii) the variance of the mean (vom)

$$q_{i,vom} = \frac{1}{N_b} \sum_{\ell=1}^{N_b} \mu_{e,i}^2(\mathbf{x}^{(\ell)}) - \bar{\mu}_{e,i}^2,$$ (32)

where $\bar{\mu}_{e,i} = \frac{1}{N_b} \sum_{\ell=1}^{N_b} \mu_{e,i}(\mathbf{x}^{(\ell)})$.

When allowing the variance to be trainable, we can compute $q_{i,vom}$ and $q_{i,mov}$ to help identify important feature directions in latent space. We show results for training the Burgers' Equation embedding the cylinder topology case of Section 3.3 in Figure 15. We see important feature components of the encoded data have the characteristic signature of having a large $q_{i,vom}$ and a small $q_{i,mov}$. If we reduce the encoding to just these components, we can obtain a lower dimensional embedding for the data. We find this embedding have a cylindrical geometric structure providing a good latent space representation of the system state information. In contrast, if we were to choose any two arbitrary latent code dimensions, such as z_1, z_2 we obtain a poor entangled representation, see Figure 15. This allows for using information from the $\sigma_{e,i}$ and $\mu_{e,i}$ to further extract geometric structure from data. This can be used to formulate latent manifold spaces for use with our GD-VAE approaches.

References

[1] Evan Archer, Il Memming Park, Lars Buesing, John Cunningham, and Liam Paninski. Black box variational inference for state space models. *arXiv preprint arXiv:1511.07367*, 2015. URL https://arxiv.org/abs/1511.07367.

[2] Georgios Arvanitidis, Lars Kai Hansen, and Søren Hauberg. Latent space oddity: on the curvature of deep generative models. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SJzRZ-WCZ.

[3] Paul J. Atzberger. Importance of the mathematical foundations of machine learning methods for scientific and engineering applications. *SciML2018 Workshop, (position paper)*, 2018. URL https://arxiv.org/abs/1808.02213.

[4] Paul J. Atzberger. Mlmod package: Machine learning methods for data-driven modeling in lammps. *arXiv*, 2021. doi: 10.48550/ARXIV.2107.14362. URL https://arxiv.org/abs/2107.14362.

[5] Omri Azencot, Wotao Yin, and Andrea Bertozzi. Consistent dynamic mode decomposition. *SIAM Journal on Applied Dynamical Systems*, 18(3):1565–1585, 2019. URL https://www.math.ucla.edu/~bertozzi/papers/CDMD_SIADS.pdf.

[6] Harry Bateman. Some Recent Researches on the Motion of Fluids. *Monthly Weather Review*, 43(4):163, January 1915. doi: 10.1175/1520-0493(1915)43⟨163:SRROTM⟩2.0.CO;2.

[7] Leonard E. Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state markov chains. *Ann. Math. Statist.*, 37(6):1554–1563, 12 1966. doi: 10.1214/aoms/1177699147. URL https://doi.org/10.1214/aoms/1177699147.

[8] Christopher M. Bishop, Markus Svensén, and Christopher K. I. Williams. GTM: A principled alternative to the self-organizing map. In Michael Mozer, Michael I. Jordan, and Thomas Petsche, editors, *Advances in Neural Information Processing Systems 9, NIPS, Denver, CO, USA, December 2-5, 1996*, pages 354–360. MIT Press, 1996. URL http://papers.nips.cc/paper/1207-gtm-a-principled-alternative-to-the-self-organizing-map.

[9] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017. doi: 10.1080/01621459.2017.1285773. URL https://doi.org/10.1080/01621459.2017.1285773.

[10] Steven L. Brunton and J. Nathan Kutz. *Reduced Order Models (ROMs)*, page 375–402. Cambridge University Press, 2019. doi: 10.1017/9781108380690.012.

[11] Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016. ISSN 0027-8424. doi: 10.1073/pnas.1517384113. URL https://www.pnas.org/content/113/15/3932.

[12] Richard L. Burden and Douglas Faires. *Numerical Analysis*. Brooks/Cole Cengage Learning, 2010.

[13] Gunnar Carlsson, Tigran Ishkhanov, Vin de Silva, and Afra Zomorodian. On the local behavior of spaces of natural images. *International Journal of Computer Vision*, 76(1):1–12, January 2008. ISSN 1573-1405. URL https://doi.org/10.1007/s11263-007-0056-x.

[14] Kathleen Champion, Bethany Lusch, J. Nathan Kutz, and Steven L. Brunton. Data-driven discovery of coordinates and governing equations. *Proceedings of the National Academy of Sciences*, 116(45):22445–22451, 2019. ISSN 0027-8424. doi: 10.1073/pnas.1906995116. URL https://www.pnas.org/content/116/45/22445.

[15] Anindya Chatterjee. An introduction to the proper orthogonal decomposition. *Current Science*, 78(7):808–817, September 2000. ISSN 00113891. URL http://www.jstor.org/stable/24103957.

[16] Nutan Chen, Maximilian Karl, and Patrick Van Der Smagt. Dynamic movement primitives in latent space of time-dependent variational autoencoders. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pages 629–636. IEEE, 2016. URL https://ieeexplore.ieee.org/document/7803340.

[17] Nutan Chen, Alexej Klushyn, Francesco Ferroni, Justin Bayer, and Patrick Van Der Smagt. Learning flat latent manifolds with VAEs. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1587–1596, Virtual, 13–18 Jul 2020. PMLR. URL http://proceedings.mlr.press/v119/chen20i.html.

[18] Alessandro Chiuso and Gianluigi Pillonetto. System identification: A machine learning perspective. *Annual Review of Control, Robotics, and Autonomous Systems*, 2(1):281–304, 2019. doi: 10.1146/annurev-control-053018-023744. URL https://doi.org/10.1146/annurev-control-053018-023744.

[19] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1179. URL https://www.aclweb.org/anthology/D14-1179.

[20] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C. Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. *Advances in neural information processing systems*, abs/1506.02216, 2015. URL http://arxiv.org/abs/1506.02216.

[21] James P. Crutchfield and Bruce S. McNamara. Equations of motion from a data series. *Complex Syst.*, 1, 1987.

[22] Suddhasattwa Das and Dimitrios Giannakis. Delay-coordinate maps and the spectra of koopman operators. 175:1107–1145, 2019. ISSN 0022-4715. doi: 10.1007/s10955-019-02272-w.

[23] Tim R. Davidson, Luca Falorsi, Nicola De Cao, Thomas Kipf, and Jakub M. Tomczak. Hyperspherical variational auto-encoders. 2018. URL https://arxiv.org/abs/1804.00891.

[24] Pierre Del Moral. Nonlinear filtering: Interacting particle resolution. *Comptes Rendus de l'Académie des Sciences - Series I - Mathematics*, 325(6):653 – 658, 1997. ISSN 0764-4442. doi: https://doi.org/10.1016/S0764-4442(97)84778-7. URL http://www.sciencedirect.com/science/article/pii/S0764444297847787.

[25] Ronald A. DeVore. *Model Reduction and Approximation: Theory and Algorithms*, chapter Chapter 3: The Theoretical Foundation of Reduced Basis Methods, pages 137–168. SIAM, 2017. doi: 10.1137/1.9781611974829.ch3. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611974829.ch3.

[26] Benjamin Erichson, Michael Muehlebach, and Michael W Mahoney. Physics-informed autoencoders for lyapunov-stable fluid flow prediction. *arXiv preprint arXiv:1905.10866*, 2019.

[27] Luca Falorsi, P. D. Haan, T. Davidson, Nicola De Cao, M. Weiler, Patrick Forré, and T. Cohen. Explorations in homeomorphic variational auto-encoding. *ArXiv*, abs/1807.04689, 2018. URL https://arxiv.org/pdf/1807.04689.pdf.

[28] Zoubin Ghahramani and Sam T. Roweis. Learning nonlinear dynamical systems using an EM algorithm. In Michael J. Kearns, Sara A. Solla, and David A. Cohn, editors, *Advances in Neural Information Processing Systems 11, [NIPS Conference, Denver, Colorado, USA, November 30 - December 5, 1998]*, pages 431–437. The MIT Press, 1998. URL http://papers.nips.cc/paper/1594-learning-nonlinear-dynamical-systems-using-an-em-algorithm.

[29] Laurent Girin, Simon Leglaive, Xiaoyu Bie, Julien Diard, Thomas Hueber, and Xavier Alameda-Pineda. Dynamical variational autoencoders: A comprehensive review. 2020.

[30] Simon Godsill. Particle filtering: the first 25 years and beyond. In *Proc. Speech and Signal Processing (ICASSP) ICASSP 2019 - 2019 IEEE Int. Conf. Acoustics*, pages 7760–7764, 2019.

[31] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. The MIT Press, 2016. ISBN 0262035618. URL https://www.deeplearningbook.org/.

[32] Ben J. Gross, Nathian Trask, Paul Kuberry, and Paul J. Atzberger. Meshfree methods on manifolds for hydrodynamic flows on curved surfaces: A generalized moving least-squares (gmls) approach. *Journal of Computational Physics*, 409:109340, 2020. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2020.109340. URL http://www.sciencedirect.com/science/article/pii/S0021999120301145.

[33] Carlos X. Hernández, Hannah K. Wayment-Steele, Mohammad M. Sultan, Brooke E. Husic, and Vijay S. Pande. Variational encoding of complex dynamics. *Physical Review E*, 97(6), Jun 2018. ISSN 2470-0053. doi: 10.1103/physreve.97.062412. URL http://dx.doi.org/10.1103/PhysRevE.97.062412.

[34] Jan S. Hesthaven, Gianluigi Rozza, and Benjamin Stamm. Reduced basis methods. pages 27–43, 2016. ISSN 2191-8198. doi: 10.1007/978-3-319-22470-1_3.

[35] Morris Hirsch and Stephen Smale. *Differential Equations, Dynamical Systems, and Linear Algebra (Pure and Applied Mathematics, Vol. 60)*. Academic Press, Inc., 1974.

[36] Morris W Hirsch, Stephen Smale, and Robert L Devaney. *Differential equations, dynamical systems, and an introduction to chaos*. Academic press, 2012.

[37] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8): 1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

[38] Xia Hong, Richard James Mitchell, Sheng Chen, C. J. Harris, Keener Li, and Gareth W. Irwin. Model selection approaches for non-linear system identification: a review. *International Journal of Systems Science*, 39(10):925–946, 2008. doi: 10.1080/00207720802083018. URL https://doi.org/10.1080/00207720802083018.

[39] Eberhard Hopf. The partial differential equation $u_t + uu_x = \mu_{xx}$. *Comm. Pure Appl. Math. 3, 201-230*, 01 1950. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/cpa.3160030302.

[40] Arieh Iserles. *A first course in the numerical analysis of differential equations*. Number 44. Cambridge university press, 2009.

[41] Kristopher T. Jensen, Ta-Chu Kao, Marco Tripodi, and Guillaume Hennequin. Manifold gplvms for discovering non-euclidean latent structure in neural data. 2020. URL https://arxiv.org/abs/2006.07429.

[42] Dimitris Kalatzis, David Eklund, Georgios Arvanitidis, and Søren Hauberg. Variational autoencoders with riemannian brownian motion priors. *arXiv e-prints*, art. arXiv:2002.05227, February 2020. URL https://arxiv.org/abs/2002.05227.

[43] Rudolph E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 03 1960. ISSN 0021-9223. doi: 10.1115/1.3662552. URL https://doi.org/10.1115/1.3662552.

[44] Wilhelm Killing. *Die nicht-euklidischen Raumformen in analytischer Behandlung*. BG Teubner, 1885.

[45] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6114.

[46] Teuvo Kohonen. Self-organized formation of topologically correct feature maps. *Biological cybernetics*, 43(1):59–69, 1982. URL https://link.springer.com/article/10.1007/BF00337288.

[47] Milan Korda, Mihai Putinar, and Igor Mezić. Data-driven spectral analysis of the koopman operator. *Applied and Computational Harmonic Analysis*, 48(2):599 – 629, 2020. ISSN 1063-5203. doi: https://doi.org/10.1016/j.acha.2018.08.002. URL http://www.sciencedirect.com/science/article/pii/S1063520318300988.

[48] Rahul G. Krishnan, Uri Shalit, and David A. Sontag. Structured inference networks for nonlinear state space models. In Satinder P. Singh and Shaul Markovitch, editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 2101–2109. AAAI Press, 2017. URL http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14215.

[49] J. Nathan Kutz, Steven L. Brunton, Bingni W. Brunton, and Joshua L. Proctor. *Dynamic Mode Decomposition*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2016. doi: 10.1137/1.9781611974508. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611974508.

[50] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015. ISSN 1476-4687. URL https://doi.org/10.1038/nature14539.

[51] Kookjin Lee and Kevin T. Carlberg. Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders. *Journal of Computational Physics*, 404:108973, 2020. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2019.108973. URL http://www.sciencedirect.com/science/article/pii/S0021999119306783.

[52] Ryan Lopez and Paul J. Atzberger. Variational autoencoders for learning nonlinear dynamics of physical systems. *arXiv:2012.03448*, 2020. doi: 10.48550/ARXIV.2012.03448. URL https://arxiv.org/abs/2012.03448.

[53] Ryan Lopez and Paul J. Atzberger. Variational autoencoders for learning nonlinear dynamics of physical systems. *AAAI Workshop Proceedings*, 2020. URL https://ceur-ws.org/Vol-2964/article_174.pdf.

[54] Peter Y Lu, Samuel Kim, and Marin Soljačić. Extracting interpretable physical parameters from spatiotemporal systems using unsupervised learning. *Physical Review X*, 10(3):031056, 2020. doi: 10.1103/PhysRevX.10.031056. URL https://link.aps.org/doi/10.1103/PhysRevX.10.031056.

[55] Bethany Lusch, J. Nathan Kutz, and Steven L. Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Communications*, 9(1):4950, November 2018. ISSN 2041-1723. URL https://doi.org/10.1038/s41467-018-07210-0.

[56] Horia Mania, Michael I Jordan, and Benjamin Recht. Active learning for nonlinear system identification with guarantees. *arXiv preprint arXiv:2006.10277*, 2020. URL https://arxiv.org/pdf/2006.10277.pdf.

[57] Miguel Alfonso Mendez, Mikhael Balabane, and Jean Marie Buchlin. Multi-scale proper orthogonal decomposition (mpod). 2018. doi: 10.1063/1.5043720.

[58] Igor Mezić. Analysis of fluid flows via spectral properties of the koopman operator. *Annual Review of Fluid Mechanics*, 45(1):357–378, 2013. doi: 10.1146/annurev-fluid-011212-140652. URL https://doi.org/10.1146/annurev-fluid-011212-140652.

[59] Oliver Nelles. *Nonlinear system identification: from classical approaches to neural networks and fuzzy models*. Springer Science & Business Media, 2013. URL https://play.google.com/books/reader?id=tyjrCAAAQBAJ&hl=en&pg=GBS.PR3.

[60] Mario Ohlberger and Stephan Rave. Reduced basis methods: Success, limitations and future challenges. *Proceedings of the Conference Algoritmy*, pages 1–12, 2016. URL http://www.iam.fmph.uniba.sk/amuc/ojs/index.php/algoritmy/article/view/389.

[61] Eric J. Parish and Kevin T. Carlberg. Time-series machine-learning error models for approximate solutions to parameterized dynamical systems. *Computer Methods in Applied Mechanics and Engineering*, 365:112990, 2020. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2020.112990. URL http://www.sciencedirect.com/science/article/pii/S0045782520301742.

[62] Suraj Pawar, Shady E. Ahmed, Omer San, and Adil Rasheed. Data-driven recovery of hidden physics in reduced order modeling of fluid flows. 32:036602, 2020. ISSN 1070-6631. doi: 10.1063/5.0002051.

[63] Michael Pearce. The gaussian process prior vae for interpretable latent dynamics from pixels. volume 118 of *Proceedings of Machine Learning Research*, pages 1–12. PMLR, 08 Dec 2020. URL http://proceedings.mlr.press/v118/pearce20a.html.

[64] Jose A. Perea and Gunnar Carlsson. A klein-bottle-based dictionary for texture representation. *International Journal of Computer Vision*, 107(1):75–97, March 2014. ISSN 1573-1405. URL https://doi.org/10.1007/s11263-013-0676-2.

[65] Luis A. Perez Rey, Vlado Menkovski, and Jim Portegies. Diffusion variational autoencoders. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 2704–2710. International Joint Conferences on Artificial Intelligence Organization, 7 2020. doi: 10.24963/ijcai.2020/375. URL https://arxiv.org/pdf/1901.08991.pdf.

[66] Ilya Prigogine and René Lefever. Symmetry breaking instabilities in dissipative systems. ii. *The Journal of Chemical Physics*, 48(4):1695–1700, 1968.

[67] Ilya Prigogine and Grégoire Nicolis. On symmetry-breaking instabilities in dissipative systems. *The Journal of Chemical Physics*, 46(9):3542–3550, 1967.

[68] Maziar Raissi and George Em Karniadakis. Hidden physics models: Machine learning of non-linear partial differential equations. *Journal of Computational Physics*, 357:125 – 141, 2018. ISSN 0021-9991. URL https://arxiv.org/abs/1708.00588.

[69] Geoffrey Roeder, Paul K. Grant, Andrew Phillips, Neil Dalchau, and Edward Meeds. Efficient amortised bayesian inference for hierarchical and nonlinear dynamical systems. 2019. URL https://arxiv.org/abs/1905.12090.

[70] Steven L. Brunton Samuel H. Rudy, J. Nathan Kutz. Deep learning of dynamics and signal-noise decomposition with time-stepping constraints. *arXiv:1808:02578*, 2018. URL https://doi.org/10.1016/j.jcp.2019.06.056.

[71] Fadil Santosa and William W Symes. Linear inversion of band-limited reflection seismograms. *SIAM journal on scientific and statistical computing*, 7(4):1307–1330, 1986.

[72] Nikolaos Sarafianos, Bogdan Boteanu, Bogdan Ionescu, and Ioannis A. Kakadiaris. 3d human pose estimation: A review of the literature and analysis of covariates. *Computer Vision and Image Understanding*, 152:1 – 20, 2016. ISSN 1077-3142. doi: https://doi.org/10.1016/j.cviu.2016.09.002. URL http://www.sciencedirect.com/science/article/pii/S1077314216301369.

[73] Lawrence K. Saul. A tractable latent variable model for nonlinear dimensionality reduction. *Proceedings of the National Academy of Sciences*, 117(27):15403–15408, 2020. ISSN 0027-8424. doi: 10.1073/pnas.1916012117. URL https://www.pnas.org/content/117/27/15403.

[74] Peter J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28, 2010. doi: 10.1017/S0022112010001217. URL https://doi.org/10.1017/S0022112010001217.

[75] Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. 324:81–85, 2009. ISSN 0036-8075. doi: 10.1126/science.1165893.

[76] Jonas Sjöberg, Qinghua Zhang, Lennart Ljung, Albert Benveniste, Bernard Delyon, Pierre-Yves Glorennec, Håkan Hjalmarsson, and Anatoli Juditsky. Nonlinear black-box modeling in system identification: a unified overview. *Automatica*, 31(12):1691 – 1724, 1995. ISSN 0005-1098. doi: https://doi.org/10.1016/0005-1098(95)00120-8. URL http://www.sciencedirect.com/science/article/pii/0005109895001208. Trends in System Identification.

[77] Steven H Strogatz. *Nonlinear dynamics and chaos: with applications to physics, biology, chemistry, and engineering.* CRC press, 2018.

[78] R. Talmon, S. Mallat, H. Zaveri, and R. R. Coifman. Manifold learning for latent variable inference in dynamical systems. *IEEE Transactions on Signal Processing*, 63(15):3843–3856, 2015. doi: 10.1109/TSP.2015.2432731.

[79] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288, 1996.

[80] Nathanial Trask, Ravi Patel, Paul Atzberger, and Ben Gross. Gmls-nets: A machine learning framework for unstructured data. In *Proceedings of the Thirty-Fourth AAAI MLPS*, 2020. URL http://ceur-ws.org/Vol-2587/article_9.pdf.

[81] Jonathan H. Tu, Clarence W. Rowley, Dirk M. Luchtenburg, Steven L. Brunton, and J. Nathan Kutz. On dynamic mode decomposition: Theory and applications. *Journal of Computational Dynamics*, 2014. URL http://aimsciences.org//article/id/1dfebc20-876d-4da7-8034-7cd3c7ae1161.

[82] Rudolph Van Der Merwe, Arnaud Doucet, Nando De Freitas, and Eric Wan. The unscented particle filter. In *Proceedings of the 13th International Conference on Neural Information Processing Systems*, NIPS'00, page 563–569, Cambridge, MA, USA, 2000. MIT Press.

[83] Klaus Volkert. Space forms: a history. *BoMA-Bulletin of the Manifold Atlas*, 2013.

[84] E. A. Wan and R. Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, pages 153–158, 2000. doi: 10.1109/ASSPCC.2000. 882463.

[85] Hassler Whitney. The self-intersections of a smooth n-manifold in 2n-space. *Annals of Mathematics*, 45(2):220–246, 1944. ISSN 0003486X. URL http://www.jstor.org/stable/1969265.

[86] Yibo Yang and Paris Perdikaris. Physics-informed deep generative models. *arXiv preprint arXiv:1812.03511*, 2018.

[87] David Zwicker. py-pde: A python package for solving partial differential equations. *Journal of Open Source Software*, 5(48):2158, 2020. doi: 10.21105/joss.02158. URL https://doi.org/10.21105/joss.02158.