Variational Autoencoding Neural Operators

Jacob H. Seidman [1 2] **Georgios Kissas** [2] **George J. Pappas** [1] **Paris Perdikaris** [2]

Abstract

Unsupervised learning with functional data is an emerging paradigm of machine learning research with applications to computer vision, climate modeling and physical systems. A natural way of modeling functional data is by learning operators between infinite dimensional spaces, leading to discretization invariant representations that scale independently of the sample grid resolution. Here we present Variational Autoencoding Neural Operators (VANO), a general strategy for making a large class of operator learning architectures act as variational autoencoders. For this purpose, we provide a novel rigorous mathematical formulation of the variational objective in function spaces for training. VANO first maps an input function to a distribution over a latent space using a parametric encoder and then decodes a sample from the latent distribution to reconstruct the input, as in classic variational autoencoders. We test VANO with different model set-ups and architecture choices for a variety of benchmarks. We start from a simple Gaussian random field where we can analytically track what the model learns and progressively transition to more challenging benchmarks including modeling phase separation in Cahn-Hilliard systems and real world satellite data for measuring Earth surface deformation.

1. Introduction

Much of machine learning research focuses on data residing in finite dimensional vector spaces. For example, images are commonly seen as vectors in a space with dimension equal to the number of pixels (Santhanam et al., 2017) and words are represented by one-hot encodings in a space representing a dictionary (Vaswani et al., 2017). Architectures that act on such data are built with this structure in mind; they aim to learn maps between finite dimensional spaces of data.

On the other hand, physics often models signals of interest in the natural world in terms of continuous fields, e.g. velocity in fluid dynamics or temperature in heat transfer. These fields are typically functions over a continuous domain, and therefore correspond to vectors in *infinite-dimensional* vector spaces, also known as functional data. To use machine learning tools for continuous signals in these physical applications, models must be able to act on and return representations of functional data.

The most straightforward way to do this is known as the discretize-first approach. Here, functional data is mapped into a finite dimensional vector space via measurements along a predefined collection of locations. At this point, standard machine learning tools for finite dimensional data can be used to generate measurements of a desired output function, also evaluated along a predefined set of locations. The drawback of these methods is their rigidity with respect to the underlying discretization scheme; they will not be able to evaluate the output function at any location outside of the original discretization.

As an alternative, operator learning methods aim to design models which give well defined operators between the function spaces themselves instead of their discretizations. These methods often take a discretization agnostic approach and are able to produce outputs that can be queried at arbitrary points in their target domain. The Graph Neural Operator (Anandkumar et al., 2020) proposed a compositional architecture built from parameterized integral transformations of the input combined with point-wise linear and nonlinear maps. This approach was modified to leverage fast Fourier transforms in computing the integral transform component, leading to the Fourier Neural Operator (Li et al., 2020), U-net variants (Wen et al., 2022), as well as guarantees of universal approximation (Kovachki et al., 2021).

Inspired from one of the first operator architectures in (Chen & Chen, 1995), the DeepONet (Lu et al., 2021) uses finite dimensional representations of input functions to derive coefficients along a learned basis of output functions. While this approach has been shown to have universal approximation properties as well (Lanthaler et al., 2022), the required

[1]Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, USA [2]Department of Mechanical Engineering and Applied Mechanics, University of Pennsylvania, Philadelphia, USA. Correspondence to: Jacob H. Seidman <seidj@sas.upenn.edu>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

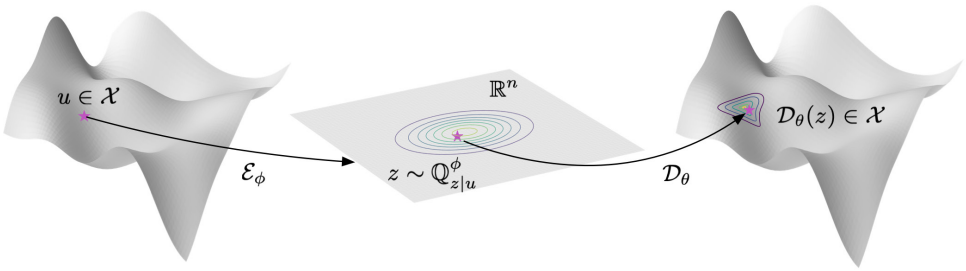

Figure 1. Schematic representation of the VANO framework: The encoder $\mathcal{E}_\phi$ maps a point from the input function manifold to a random point sampled from a variational distribution $\mathbb{Q}^\phi_{z|u}$ which is then mapped to a point on the output function manifold using the decoder $\mathcal{D}_\theta$.

size of the architecture can scale unfavorably due to the linear nature of the output function representations (Lanthaler et al., 2022; Seidman et al., 2022). Under the assumption that the output functions concentrate along a finite dimensional manifold in the ambient function space, (Seidman et al., 2022) proposed a model which builds nonlinear parameterizations of output functions and circumvents the limitations of purely linear representations.

While much recent work has focused on designing methods for functional data in a supervised setting, less has been done for unsupervised learning. Here we focus on two key aspects of unsupervised learning, namely dimensionality reduction and generative modeling. For dimensionality reduction of functional data living in a Hilbert space, the inner product structure allows for generalizations of principal components analysis (PCA) (Wang et al., 2016), also known as proper orthogonal decomposition (Chatterjee, 2000). Kernel tricks can also be employed on functional data to obtain nonlinear versions of PCA in feature spaces (Song & Li, 2021). A generalization of the manifold learning (Nadler et al., 2006) approach was taken in (Du et al., 2021) to learn distance preserving and locally linear embeddings of functional data into finite dimensional spaces.

Generative modeling of functional data has been approached by defining stochastic processes with neural networks, dubbed neural processes (Garnelo et al., 2018b;a; Kim et al., 2018). Adversarial generative models for continuous images trained on point-wise data have also been proposed in (Skorokhodov et al., 2021) and (Dupont et al., 2021), while a variational autoencoder (VAE) approach with neural radiance fields (NeRFs) was taken in (Kosiorek et al., 2021).

These methods formulate their training objective in terms of point-wise measurements, resulting in models which learn to maximize the probability of observing a collection of points and not a function itself. This makes the function sampling density play a decisive role in how well the model performs; if high and low resolution data coexist in a dataset, the model will over-fit the high resolution data (Rahman et al., 2022a). Recently, the U-NO architecture (Rahman et al., 2022b) was used in the Generative Adversarial Neural Operator (GANO) framework (Rahman et al., 2022a) to build a generator acting on samples of Gaussian random fields and a functional discriminator in order to overcome the aforementioned drawbacks.

In this paper, we provide a novel method of using encoder-decoder operator learning architectures for dimensionality reduction and generative modeling of functional data. Under the manifold hypothesis for functional data (Seidman et al., 2022), if we train these models to learn the identity map then the latent space attempts to learn finite dimensional coordinates for the data manifold. In Figure 1 we give a visual representation of our approach. We will additionally view the data as coming from a generative model on this coordinate space and train the corresponding operator learning architecture via an auto-encoding variational Bayes approach. The decoder component of the architecture then creates functions from samples in the latent coordinate space which can be queried at any point along their domain. To overcome over-fitting pathologies at higher data resolutions we describe the variational objective in a *discretization agnostic way* by putting forth a well-defined formulation with respect to the functional data space instead of spaces of point-wise evaluations.

Our main contributions can be summarized as:

- We provide the first rigorous mathematical formulation of a variational objective which is completely discretization agnostic and easily computable.
- Using this objective, we give a novel formulation of a variational autoencoder for functional data with operator learning architectures.
- We perform zero-shot super-resolution sampling of functions describing complex physical processes and real world satellite data.
- We demonstrate state-of-the-art performance in terms

of reconstruction error and sample generation quality while taking a fraction of the training time and model size compared to competing approaches.

Outline of Paper: The remainder of the paper will be structured as follows. First we will describe the class of encoder-decoder operator learning architectures and describe how they perform dimensionality reduction on functional data. Next, we will briefly review the VAE formulation for finite dimensional data before giving a mathematically precise generalization to functional data and describing the components of our model. Section 6 will present several experiments illustrating the effectiveness of this approach.

Notation We denote our functional data by $u \in \mathcal{X}$, where $\mathcal{X}$ is a function space over some domain $X \subset \mathbb{R}^d$. Typically we will take $\mathcal{X} = L^2(X)$ or $\mathcal{X} = H^s(X)$, where $H^s(X)$ is the Sobolev space of index s over X. Probability measures will be denoted by blackboard bold typefaced letters $\mathbb{P}$, $\mathbb{Q}$, and $\mathbb{V}$.

2. Encoder-Decoder Neural Operators

A large class of architectures built to learn mappings between spaces of functions $\mathcal{X} \to \mathcal{Y}$ make use of a finite dimensional latent space in the following way. First, an encoding map $\mathcal{E} : \mathcal{X} \to \mathbb{R}^n$ is learned from the input functions to a n-dimensional latent space. Then, the latent code corresponding to a functional input, $z = \mathcal{E}(u)$ is mapped to a queryable output function via a decoding map $\mathcal{D} : \mathbb{R}^n \to \mathcal{Y}$, $f(\cdot) = \mathcal{D}(z)$. For example, in the DeepONet architecture (Lu et al., 2021), input functions u are mapped via a "branch network" to a hidden vector $z \in \mathbb{R}^n$, which is then used as coefficients of a learned basis to reconstruct an output function. These methods can be interpreted as giving a finite dimensional parameterization to the set of output functions, where the parameters for a fixed output function are determined from the corresponding input function.

2.1. Linear versus Nonlinear Decoders

It was shown in (Seidman et al., 2022) that when these kinds of architectures build output functions in a linear manner from the latent space, such as in (Kissas et al., 2022; Lu et al., 2021; Bhattacharya et al., 2021), they may miss low dimensional nonlinear structure in the set of output functions that can otherwise be captured by a nonlinear map from the latent space to the output function space. The authors further gave an interpretation of this architecture under the assumption that the distribution of output functions concentrates on a low dimensional manifold in its ambient function space. In this setting, the decoder map ideally would learn a coordinate chart between the finite dimensional latent space and the manifold of output functions. This suggests that successful architectures are implicitly performing dimensionality reduction on the set of output functions.

2.2. Dimensionality Reduction through the Latent Space

We will follow this interpretation to create a natural extension of encoder-decoder operator learning architectures for dimensionality reduction and generation of functional data. If the input and output function spaces are the same $\mathcal{X} = \mathcal{Y}$ and we learn the identity map on our data factored through a finite dimensional latent space, then the encoding map $\mathcal{E} : \mathcal{X} \to \mathbb{R}^n$ gives a lower dimensional representation of our functional data. That is, when trained to approximate the identity map, these architectures become *functional autoencoders*.

If we additionally would like to generate new samples of our functional data with this framework, it would suffice to learn a probability measure over the latent space corresponding to the finite dimensional embedding of our data. Similar to the non-functional data case, this can be modelled through the use of a Variational Auto-Encoder (VAE) (Kingma & Welling, 2014), which takes a Bayesian approach to determining latent representations of observed data. While this method has been studied extensively on finite dimensional data, its extension to functional data has only been explored specifically for neural radiance fields in (Kosiorek et al., 2021), where the variational objective is formulated in terms of point-wise measurements.

In this paper, we will place operator learning architectures with a finite dimensional latent space, such as DeepONet (Lu et al., 2021) and NOMAD (Seidman et al., 2022), within the formalism of autoencoding variational Bayesian methods to simultaneously obtain a new method of dimensionality reduction and generative modeling for functional data. To do so, we must be careful to reformulate the VAE objective in function spaces. Variational objectives have been formulated considering latent spaces as function spaces, as in (Wild & Wynne, 2021; Wild et al., 2022), but a variational objective where the likelihood term is described in a functional data space has not yet been addressed.

As we will see, while the immediate application of the formulation for finite dimensional data does not apply, there exists an appropriate generalization which is mathematically rigorous and practically well behaved. The benefit of the function space formulation is the lack of reference to a particular choice of discretization of the data, leading to a more flexible objective which remains valid under different measurements of the functional data.

3. VAEs for Finite Dimensional Data

Here we review a simple generative model for finite dimensional data and the resulting variational Bayesian approach

to inference in the latent space. For this subsection only, we will define our data space as $\mathcal{X} = \mathbb{R}^d$. As before, let the latent space be $\mathcal{Z} = \mathbb{R}^n$, often with $n << d$.

Assume the following generative model for samples of u from its probability measure $\mathbb{P}_u$ on $\mathcal{X}$. Let $\mathbb{P}_z$ be a *prior* probability measure on $\mathcal{Z}$, $\mathcal{D} : \mathcal{Z} \to \mathcal{X}$ a function from the latent space to the data space, and η a noise vector sampled from a probability measure $\mathbb{V}$ on $\mathcal{X}$ such that

$$u = \mathcal{D}(z) + \eta, \quad z \sim \mathbb{P}_z, \quad \eta \sim \mathbb{V}, \tag{1}$$

is distributed according to $\mathbb{P}_u$.

According to this model, there exists a joint probability measure $\mathbb{P}$ on $\mathcal{Z} \times \mathcal{X}$ with marginals $\mathbb{P}_z$ and $\mathbb{P}_u$ as defined above. Assume these three measures have well defined probability density functions, $p(z, u)$, $p(z)$ and $p(u)$, respectively, and conditional densities $p(z|u)$ and $p(u|z)$.

Under full knowledge of these densities, we can form a low dimensional representation of a given data point u by sampling from $p(z|u)$. However, in general, the evaluation of the conditional density is intractable. This motivates the variational approach (Kingma & Welling, 2014), where we instead create a parameterized family of distributions $q^\phi(z|u)$, and attempt to approximate the true conditional $p(z|u)$ with $q^\phi(z|u)$. When the KL divergence is used as a quality of approximation from $q^\phi(z|u)$ to $p(z|u)$, this can be approached with the following optimization problem

$$\underset{\phi}{\text{minimize}} \quad \mathop{\mathbb{E}}_{u\sim p(u)} \left[\mathrm{KL}[q^\phi(z|u) \,||\, p(z|u)]\right]. \tag{2}$$

Since we do not have access to $p(z|u)$ and cannot evaluate the KL divergence term above, we optimize instead a quantity known as the *Evidence Lower Bound* (ELBO),

$$\mathcal{L} = - \mathop{\mathbb{E}}_{z\sim q^\phi(z|u)} [\log p(u|z)] + \mathrm{KL}[q^\phi(z|u) \,||\, p(z)], \tag{3}$$

which differs from the objective in (2) by a data-dependent constant,

$$\mathrm{KL}[q^\phi(z|u) \,||\, p(z|u)] = -\mathcal{L} + \log p(x) \tag{4}$$

When the prior $p(z)$ is a Gaussian and the variational distribution $q^\phi(z|u)$ is also Gaussian with a mean and variance dependent on the input u through a parameterized encoding map $\mathcal{E}_\phi(u) = (\mu_\phi(u), \sigma_\phi(u)))$, the KL term in (3) has a closed form.

Under the assumption that the noise vector in (1) is a centered Gaussian with isotropic covariance, $\eta \sim \mathcal{N}(0, \delta^2 \mathrm{I})$, the log likelihood term is equal to

$$\log p(u \mid z) = \log(2\pi\delta^2)^{-d/2} - \frac{1}{2\delta^2}\|u - \mathcal{D}(z)\|_2^2. \tag{5}$$

By parameterizing the function $\mathcal{D}$ as well, we arrive at an objective function that can be used to train the encoder $\mathcal{E}_\phi$ and decoder $\mathcal{D}_\theta$ in an end-to-end fashion.

4. VAEs for Functional Data

We begin formulating a VAE in this case analogously to the previous section; we posit the generative model in (1) which induces a joint measure $\mathbb{P}$ on $\mathcal{Z} \times \mathcal{X}$ with marginals $\mathbb{P}_u$ and $\mathbb{P}_z$. Under mild assumptions on the spaces $\mathcal{Z}$ and $\mathcal{X}$ (such as being a separable Banach spaces), there exist *regular conditional measures* $\mathbb{P}_{z|u}$ and $\mathbb{P}_{u|z}$ which are well defined $\mathbb{P}_u$-a.e. and $\mathbb{P}_z$-a.e., respectively.

At this point the formulation begins to diverge from the finite dimensional case. In an infinite dimensional function space, such as $\mathcal{X}$, we no longer have a canonical notion of a probability density function to formulate our objective function and ELBO. In particular, the first term of (3) is no longer well defined as written. We will instead reason in terms of the probability measures $\mathbb{P}_u$, $\mathbb{P}_z$, $\mathbb{P}_{z|u}$, $\mathbb{P}_{u|z}$, the variational family of measures $\mathbb{Q}^\phi_{z|u}$, the noise process measure $\mathbb{V}$, and various Radon-Nikodym derivatives between them. Proceeding in this manner we are able to derive the appropriate generalization of (3) for data in the function space $\mathcal{X}$.

Theorem 4.1. *Let $\mathcal{X}$ and $\mathcal{Z}$ be Polish spaces. Given the generative model* (1)*, assume that the conditional measure $\mathbb{P}_{u|z}$ is absolutely continuous with respect to the noise measure $\mathbb{V}$. Then the following holds*

$$\mathrm{KL}[\mathbb{Q}^\phi_{z|u} \,||\, \mathbb{P}_z] = -\mathcal{L} + \log \frac{\mathrm{d}\mathbb{P}_u}{\mathrm{d}\mathbb{V}}. \tag{6}$$

with

$$\mathcal{L} = - \mathop{\mathbb{E}}_{z\sim\mathbb{Q}^\phi_{z|u}} \left[\log \frac{d\mathbb{P}_{u|z}}{d\mathbb{V}}(u)\right] + \mathrm{KL}[\mathbb{Q}^\phi_{z|u} \,||\, \mathbb{P}_z] \tag{7}$$

Proof. The proof is provided in Appendix 4.1. □

The benefit of this formulation is that the objective function makes no reference to any particular choice of discretization or available function measurements. In this sense, it is a training objective that is truly defined on a function space; whatever measurements are available will be used to approximate this objective, ensuring a form of consistency over varying discretization schemes.

4.1. Computing the ELBO Objective

To compute the likelihood term in (7), first note that under the generative model (1), given $z \in \mathcal{Z}$ the conditional measure $\mathbb{P}_{u|z}$ corresponds to a shifted version of the noise process centered at $\mathcal{D}(z)$. The Radon-Nikodym derivative $\frac{\mathrm{d}\mathbb{P}_{u|z}}{\mathrm{d}\mathbb{V}}$ then represents the change of measure of $\mathbb{V}$ under a shift by $\mathcal{D}(z)$.

In this work we will assume that $\mathbb{V}$ is a *Gaussian* measure on the space $\mathcal{X}$. Changes of measure for translated Gaussian measures are well understood and are described by the

Cameron-Martin formula (see Appendix A); this will be the main tool which allows us to evaluate the first term in (7).

In particular, we will take η to be a pure white noise process and $\mathbb{V}$ the corresponding white noise measure. Note that this implies that our measured signals must live in the dual Sobolev space $\mathcal{X} = H^{-s}(X)$ for any $s > d/2$ (Lasanen et al., 2018). In this case, the Cameron-Martin formula gives

$$\log \frac{\mathrm{d}\mathbb{P}_{u|z}}{\mathrm{d}\mathbb{V}}(u) = -\frac{1}{2}\|\mathcal{D}(z)\|_{L^2}^2 - \langle \mathcal{D}(z), u\rangle^{\sim}, \quad (8)$$

where $\langle D(z), u\rangle^{\sim}$ can be thought of as the inner product on $L^2(X)$ extended to $H^{-s}(X)$ in the second argument and is well defined a.e. with respect to the noise process $\mathbb{V}$. Given sensor measurements of u, we can approximate this second term as the standard inner product with the corresponding measurements of $\mathcal{D}(z)$. For more details on Gaussian measures in Banach spaces, white noise, and the Cameron-Martin formula see Appendix A.

Note that the expression in (8) is the same as

$$-\frac{1}{2}\|\mathcal{D}(z)-u\|_{L^2}^2 = -\frac{1}{2}\|\mathcal{D}(z)\|_{L^2}^2 + \langle \mathcal{D}(z), u\rangle - \frac{1}{2}\|u\|_{L^2}^2,$$

except for the last term. This is what we would expect to see when our data is not functional and lies in $\mathbb{R}^d$ with a Gaussian likelihood, but since u is drawn from a shifted white noise measure it is not in L^2. However, we see that the expression we derived instead for the likelihood is the same as what we would like to use up to the model independent term $\|u\|_{L^2}^2$. In this sense, the white noise likelihood formulation of the ELBO is the natural extension of the Gaussian likelihood from the finite dimensional case.

5. Variational Autoencoding Neural Operators

Given Theorem 4.1 we can define the full Variational Autoencoding Neural Operator (VANO) after making choices for the encoding and decoding maps. See Figure 1 for a visual illustration of the overall architecture.

Encoder: The encoder will map a function $u \in \mathcal{X}$ to the probability measure $\mathbb{Q}_{z|u}^{\phi}$ on the latent space $\mathbb{R}^n$. We choose the variational family $\mathbb{Q}_{z|u}^{\phi}$ to be multivariate Gaussians with diagonal covariance. It then suffices that the encoding map takes as input the function u, and returns a mean $\mu(u) \in \mathbb{R}^n$ and n positive scalars $\sigma_1, \ldots, \sigma_n = \sigma \in \mathbb{R}^n$ to parameterize this Gaussian. Hence, we define the encoder as a map $\mathcal{E}^{\phi} : \mathcal{X} \to \mathbb{R}^n \times \mathbb{R}^n$. In this paper, we will use architectures which pass measurements of the input function u through a neural network of fixed architecture. These measurements can either be point-wise, as we take to be the case in this paper, but could also be projections onto sets of functions such as trigonometric polynomials, wavelets, or other parameterized functions.

Decoder: The decoder will take a sample z of a probability measure on the latent space $\mathbb{R}^n$ and map it to a function $\mathcal{D}(z) \in \mathcal{X}$ that can be queried at any point. In this paper, we will parameterize decoders by defining a neural network which takes in points in the domain of the functions in $\mathcal{X}$, and condition its forward pass on the latent variable z. Here we will use two main variations of this conditioning process: linear conditioning, and concatenation conditioning (see Appendix C.2 for details).

Evaluating ELBO for Training: Given a data-set of N functions $\{u^i\}_{i=1}^N$, we train VANO by optimizing the objective function

$$\mathcal{L}(\phi,\theta) = \frac{1}{N}\sum_{i=1}^{N}\Big[\mathbb{E}_{\mathbb{Q}_{z|u^i}^{\phi}}[\frac{1}{2}\|\mathcal{D}_\theta(z)\|_{L^2}^2 - \langle \mathcal{D}_\theta(z), u^i\rangle^{\sim}] + \mathrm{KL}[\mathbb{Q}_{z|u^i}^{\phi} \,||\, \mathbb{P}_z]\Big]. \quad (9)$$

The expectation over the posterior $\mathbb{Q}_{z|u^i}^{\phi}$ is approximated via Monte-Carlo by sampling S latent variables $z \sim \mathbb{Q}_{z|u^i}^{\phi}$ and computing an empirical expectation. The reparameterization trick (Kingma & Welling, 2014) is used when sampling from $\mathbb{Q}_{z|u^i}^{\phi}$ to decouple the randomness from the parameters of the encoder $\mathcal{E}^{\phi}$ and allow for the computation of gradients with respect to the parameters ϕ. As $\mathcal{D}(z)$ can be evaluated at any point in the domain X, we can approximate the terms inside this expectation with whichever measurements are available for the data u^i. For example, with point-wise measurements $u(x_1), \ldots, u(x_m)$ we can use the approximation

$$\langle \mathcal{D}_\theta(z), u^i\rangle^{\sim} \approx \sum_{i=1}^{m} \mathcal{D}_\theta(x_i)u(x_i). \quad (10)$$

To avoid pathologies in the optimization of (9), we train our models with a scalar hyper-parameter β in front of the KL divergence term as in (Higgins et al., 2017) to balance the interplay between the KL and reconstruction losses.

6. Experiments

In this section we consider four examples for testing the performance of our model. In the first example, we learn the distribution corresponding to a Gaussian random field (GRF). We show that a model with a linear decoder architecture is able to accurately recover the Karhunen-Loéve decomposition (Adler, 1990), which is known to be an L^2 optimal dimension reduced approximation of the true field. Next, we examine the impact of different decoder architectures for learning distributions which do not immediately concentrate on low dimensional linear spaces, using a functional data-set of bivariate Gaussian pdfs. Next, we learn

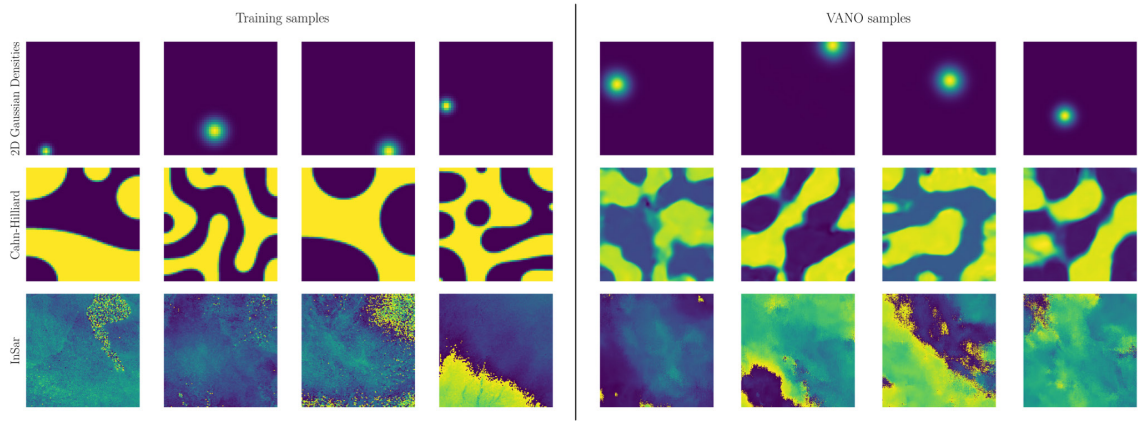

Figure 2. Representative function samples from the different benchmarks considered in this work. Left: Example functions from the testing data-sets, Right: Super-resolution samples generated by VANO.

solutions to a Cahn-Hilliard partial differential equation (PDE) system representing patterns from a phase separation processes in binary mixtures. Finally, we employ the real world InSAR interferogram data-set presented in (Rahman et al., 2022a) to demonstrate state-of-the-art performance compared to recent operator learning methods for generative modeling. For the first three experiments, we use a MMD metric for functional data (Wynne & Duncan, 2022) between samples from the different models and the ground truth to assess performance. For the last example we compare angular statistics between model generated samples and the ground truth. As an overview, Figure 2 shows some samples generated by our model on three of the data-sets we consider. More details on the metrics, hyper-parameters, architectures and the training set-up can be found in the Appendix in Section D.

6.1. Gaussian Random Field

The motivation of this example is to study the quality of the reduced dimension approximation that our model learns. For this purpose, we aim to learn a zero mean Gaussian random field (GRF) on $X = [0, 1]$ with zero boundary conditions and covariance operator $\Gamma = (\mathrm{I} - \Delta)^{-\alpha}$. This operator admits the orthonormal eigendecomposition

$$\Gamma = \sum_{i=1}^{\infty} \lambda_i \varphi_i \otimes \varphi_i,$$

where $\lambda_i = ((2\pi i)^2 + \tau^2)^{-\alpha}$ and $\varphi_i(x) = \sqrt{2}\sin(2\pi i x)$. From the Karhunen-Loéve theorem (Adler, 1990) we can construct random functions distributed as

$$u = \sum_{i=1}^{\infty} \xi_i \sqrt{\lambda_i} \varphi_i, \tag{11}$$

where $\xi_i \sim \mathcal{N}(0, 1)$ are normally distributed random variables.

We use the above sum truncated at 32 eigenpairs to construct a data-set of N functions $\{u^i\}_{i=1}^N$ and use it to train a Variational Autoencoding Neural Operator with a linear decoder. By setting the prior $\mathbb{P}_z$ to be a standard Gaussian on $\mathbb{R}^n$, a linear decoder $\mathcal{D}$ learns basis functions $\tau_i \in \mathcal{X}$ which map samples from the prior to functions,

$$\mathcal{D}(z)(x) = \sum_{i=1}^{n} z_i \tau_i(x), \quad z_i \sim \mathcal{N}(0, 1), \quad \text{i.i.d.}$$

The Karhunen Loéve theorem again tells us that the optimal choice of decoder basis functions τ_i should be exactly the eigenfunctions φ_i of the covariance operator Γ scaled by $\sqrt{\lambda_i}$. To evaluate the model performance we compare the covariance operators between the true Gaussian random field and the learned model (6.1) using a normalized Hilbert-Schmidt norm, $\|\Gamma - \hat{\Gamma}\|_{HS}^2 / \|\Gamma\|_{HS}^2$.

We present the values of the normalized Hilbert-Schmidt norm for different latent dimension sizes and over multiple model initializations in Figure 3. The right side of Figure 3 shows that the learned basis functions align closely with the optimal choice from the Karhunen Loéve theorem, scaled eigenfunctions $\sqrt{\lambda_i}\phi_i$. Generated samples from the trained model are depicted in Figure 6 in the Appendix.

6.2. The Need for Nonlinear Decoders

In this example we examine the effect of using linear versus nonlinear decoders for learning distributions of functional data. We construct a data-set consisting of bivariate Gaussian density functions over the unit square $[0, 1]^2$ where the mean is sampled randomly within the domain and the covariance is a random positive multiple of the identity. Perform-

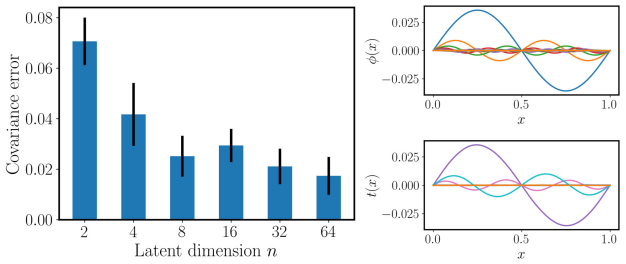

Figure 3. Left: Normalized Hilbert-Schmidt norm error between GRF samples generated from VANO and the ground truth for different sizes of the latent space. Right: Comparison between the optimal basis from the Karhunen Loéve theorem (top) and the learned basis (bottom).

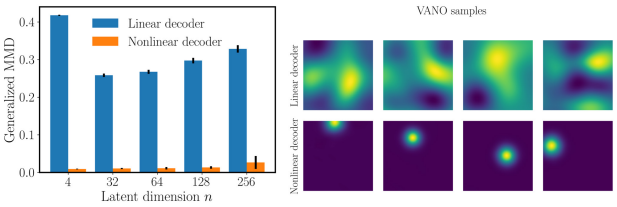

Figure 4. Left: Generalized MMD values for VANO with a linear and a nonlinear decoder for different latent dimensions, Right: Generated samples from VANO with a linear (top) and with a nonlinear decoder (bottom).

ing PCA on this data-set shows a spectrum of eigenvalues with slow decay, indicating that architectures with linear decoders will not be able to capture these functional data unless their hidden dimension is very large (see Appendix figure 7). To test this, we train VANO models with a linear and nonlinear decoder, respectively, over a range of hidden dimensions. We measure the distance of the learned distribution of functions to the ground truth via the generalized MMD distance, see Appendix E.2.

In the left panel of Figure 4 we present values of the MMD metric over five individual runs and samples generated from VANO with a linear and nonlinear decoder. We observe that the error metric for the linear model takes very high values even at larger latent dimensions. In the right panel we see representative samples generated from linear and nonlinear decoder versions of VANO. The linear decoder is not able to localize the 2D functions as in the data-set, while the nonlinear decoder produces very similar samples. In Figures 8 and 9 of the Appendix we show the reconstruction of randomly chosen cases from the test data-set for the linear and nonlinear decoders, respectively. More samples from the linear and nonlinear decoder versions of VANO are shown in Figures 11 and 10 of the Appendix.

Table 1. **Varying Training Resolutions:** Generalized MMD distance between ground truth test samples and samples generated by different models. Both VANO and the discretize-first VAE are trained and tested on 64×64, 128×128, and 256×256 resolution data.

	VANO	VAE
64×64	7.19e-03 $\pm$ 3.66e-04	7.77e-03 $\pm$ 4.38e-04
128×128	6.95e-03 $\pm$ 4.37e-04	6.39e-03 $\pm$ 1.18e-04
256×256	6.82e-03 $\pm$ 2.50e-04	6.38e-03 $\pm$ 3.61e-04

6.3. Phase Separation Patterns in Cahn-Hilliard Systems

As a more challenging benchmark we consider the Cahn-Hilliard patterns data-set (Kobeissi & Lejeune, 2022) which contains different patterns derived from the solution of the Cahn-Hilliard equation. The Cahn-Hilliard equation is a fourth-order partial differential equation that describes the evolution of the phase separation process in binary material mixtures, see Appendix Section D.3, for details.

Here we compare the VANO model to a discretize-first convolutional VAE approach. Both models are trained and tested on on Cahn-Hilliard patterns at resolutions 64x64, 128x128, and 256x256. We present the results of the generalized MMD metric for each model in Table 1. We observe that the VANO model performs better than the VAE at resolution 64x64 while the VAE achieves slightly smaller GMMD compared to VANO at higher resolutions. We attribute this to the discontinuous nature of the target functions in this benchmark in conjunction with the bias of the VANO MLP decoders toward smooth functions (Rahaman et al., 2019). However, we emphasize that the benefit of the function space formulation over the traditional discretize-first approach is in its ability to generate samples of different resolutions without any additional training or interpolation.

To showcase this ability, we train both VANO and the discretize-first VAE on 64x64 resolution data and use the trained models to generate samples of all resolutions. As the discretize-first VAE can only generate samples at the original 64x64 resolution, we generate samples of higher resolution through a bilinear interpolation of the 64x64 generated samples. For VANO, samples of higher resolution can be generated simply by evalutating the generated function samples at additional query locations. In Table 2 we see that even though VANO was only trained at the low resolution, it is still able to produce high resolution samples with a small generalized MMD to the ground truth data, while the generalized MMD of the interpolations from the discretize-first VAE increases significatnly with larger resolutions. This highlights the super-resolution capabilities of using models designed to output true functional data. Additionally, we present functions sampled from both models in

Table 2. **Super-resolution sample generation from 64x64 resolution training:** Generalized MMD distance between ground truth test samples and samples generated by different models. Both VANO and the discretize-first VAE are trained on 64×64 resolution data. Data is then generated at higher resolutions for testing either directly from the model (VANO) or through interpolation of samples generated at 64×64 resolution (VAE).

	VANO	VAE
64×64	7.19e-03 $\pm$ 3.66e-04	7.77e-03 $\pm$ 4.38e-04
128×128	8.62e-03 $\pm$ 3.66e-04	1.06e-02 $\pm$ 3.59e-04
256×256	9.44e-03 $\pm$ 3.83e-04	1.15e-02 $\pm$ 3.40e-04

the Appendix Section D.4, as well as reconstructions from each model at different resolutions. In Figure 13 we see that at higher resolutions the samples generated by the VANO model have smoother boundaries and appear more natural than those created by the discretize-first VAE approach.

Finally, we compare VANO against a recently proposed neural operator based GAN, the Generative Adversarial Neural Operator (GANO) (Rahman et al., 2022a). We train both models on 128x128 resolution Cahn-Hilliard data and find that the GANO achieves a generalized MMD of 4.88e-02 $\pm$ 4.02e-03, while VANO achieves a generalized MMD of 1.05e-02 $\pm$ 3.40e-04. In Figure 14 of Appendix D.3 we present a random selection of generated samples from each model.

6.4. Interferometric Synthetic Aperture Radar data-set

As a final example, we consider the data-set proposed by Rahman et. al. (Rahman et al., 2022a) consisting of Interferometric Synthetic Aperture Radar (InSAR) data. InSAR is a sensing technology that exploits radar signals from aerial vehicles to measure the deformation of the Earth surface for studying the dilation of volcanoes, earthquakes or underwater reserves. As a comparison, we use the Generative Adversarial Neural Operator (GANO) architecture and training parameters provided in (Rahman et al., 2022a).

We train VANO on the entire data-set using the set-up provided in the Appendix Section D.4. We evaluate the performance of our model using two metrics: circular variance and circular skewness. These are moments of angular random variables, see (Rahman et al., 2022a), used to evaluate the quality of the generated functions. In Figure 5 we present a comparison between the circular statistics (Rahman et al., 2022a), see Appendix Section E.3 for details on the metrics, for $N = 4096$ samples from the true data-set, and those created from VANO and GANO. In Section D.4 we present samples generated from both models. We observe that the VANO model achieves superior performance both in terms of circular statistics metrics, as well as in generating realistic samples without spurious artifacts. In Figure 15 we present sample reconstructions of the data from VANO and see that it also acts as a denoiser for the original data. Moreover, we find that VANO is trains 4x faster, with the 1/4 of model size compared to GANO (see Appendix Tables 4 and 5).

7. Discussion

In this work, we have shown that a large class of architectures designed for supervised operator learning can be modified to behave as variational auto-encoders for functional data. The performance of this approach was demonstrated through learning generative models for functional data coming from synthetic benchmarks, solutions of PDE systems, and real satellite data. By deriving an appropriate variational objective in an (infinite-dimensional) functional data space, we placed this approach on firm mathematical footing.

These models inherit some limitations common to all VAE approaches. In particular, there is a constant tension in the objective function of balancing the reconstruction loss of the data and the distance of the variational posterior distribution to the prior. The additional scalar parameter β multiplying the KL divergence term (Higgins et al., 2017) attempts to provide some control of this balance, but the performance of the model can be sensitive to the setting of this parameter. Some insight can be gained into controlling this phenomenon through rate distortion theory (Burgess et al., 2018), and it is an interesting direction of future work to generalize this to functional data. The choice of prior distribution on the latent space can also have a large impact on model performance. Hierarchical priors could provide additional structure for the generative model, as has been shown in (Vahdat & Kautz, 2020). We note that as the VANO models presented here use finite dimensional latent spaces, they also inherit from VAEs the ability to do interpolations in this latent space.

The approach presented in this paper bears some similarity to generative models built for learning neural fields in computer vision (Chen & Zhang, 2019; Anokhin et al., 2021). Our theoretical foundation of a variational lower bound for functional data can be directly applied to these approaches instead of the typical formulation of a likelihood on fixed point-wise measurements. This similarity also points to a larger connection between operator learning methods and conditioned neural fields in vision applications (Xie et al., 2022). Both approaches aim to build neural representations of functions which can be queried at arbitrary points of their domains and the techniques developed to do so are likely to be useful across both domains.

Finally, adapting the conditional version of the variational objective (Sohn et al., 2015) can be useful for supervised operator learning where there is aleatoric uncertainty in the output functions. For example, this can often be the

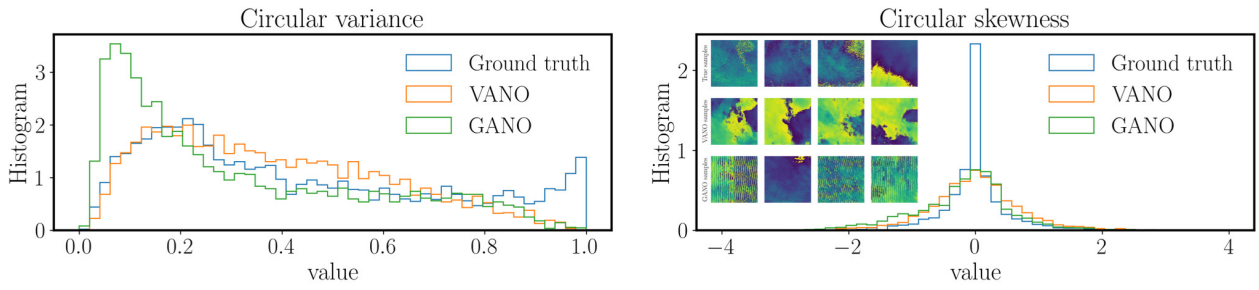

Figure 5. Left: Circular variance of the true data-set, the VANO generated data and the GANO generated data. Right: Circular skewness and generated samples between the true data-set, the VANO generated data and the GANO generated data.

case in inverse problems where not all functional outputs of a system are available for observation. Applying the variational approach put forth in this paper would give a natural notion of uncertainty quantification, while retaining the approximation capabilities of encoder-decoder operator learning architectures.

Acknowledgements

We would like to acknowledge support from the US Department of Energy under under the Advanced Scientific Computing Research program (grant DE-SC0019116), the NSF-Simmons Mathematics of Deep Learning (THEORINET), and the US Air Force Office of Scientific Research (grant AFOSR FA9550-20-1-0060). We also thank the developers of the software that enabled our research, including JAX (Bradbury et al., 2018), Matplotlib (Hunter, 2007), Pytorch (Paszke et al., 2019) and NumPy (Harris et al., 2020).

References

Adler, R. J. An introduction to continuity, extrema, and related topics for general Gaussian processes. *Lecture Notes-Monograph Series*, 12:i–155, 1990.

Anandkumar, A., Azizzadenesheli, K., Bhattacharya, K., Kovachki, N., Li, Z., Liu, B., and Stuart, A. Neural operator: Graph kernel network for partial differential equations. In *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*, 2020.

Anokhin, I., Demochkin, K., Khakhulin, T., Sterkin, G., Lempitsky, V., and Korzhenkov, D. Image generators with conditionally-independent pixel synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14278–14287, 2021.

Bhattacharya, K., Hosseini, B., Kovachki, N. B., and Stuart, A. M. Model reduction and neural networks for parametric PDEs. *The SMAI journal of computational mathematics*, 7:121–157, 2021.

Bogachev, V. I. *Gaussian Measures*, volume 62. American Mathematical Soc., 2015.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Burgess, C. P., Higgins, I., Pal, A., Matthey, L., Watters, N., Desjardins, G., and Lerchner, A. Understanding disentangling in β-VAE. *CoRR*, abs/1804.03599, 2018. URL http://arxiv.org/abs/1804.03599.

Chatterjee, A. An introduction to the proper orthogonal decomposition. *Current Science*, pp. 808–817, 2000.

Chen, T. and Chen, H. Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. *IEEE Transactions on Neural Networks*, 6(4):911–917, 1995.

Chen, Z. and Zhang, H. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5939–5948, 2019.

Di Leoni, P. C., Lu, L., Meneveau, C., Karniadakis, G. E., and Zaki, T. A. Neural operator prediction of linear instability waves in high-speed boundary layers. *Journal of Computational Physics*, 474:111793, 2023.

Du, Y., Collins, K., Tenenbaum, J., and Sitzmann, V. Learning signal-agnostic manifolds of neural fields. *Advances in Neural Information Processing Systems*, 34: 8320–8331, 2021.

Dupont, E., Teh, Y. W., and Doucet, A. Generative models as distributions of functions. *arXiv preprint arXiv:2102.04776*, 2021.

Fukumizu, K., Gretton, A., Lanckriet, G., Schölkopf, B., and Sriperumbudur, B. K. Kernel choice and classifiability for RKHS embeddings of probability distributions. *Advances in neural information processing systems*, 22, 2009.

Garnelo, M., Rosenbaum, D., Maddison, C., Ramalho, T., Saxton, D., Shanahan, M., Teh, Y. W., Rezende, D., and Eslami, S. A. Conditional neural processes. In *International Conference on Machine Learning*, pp. 1704–1713. PMLR, 2018a.

Garnelo, M., Schwarz, J., Rosenbaum, D., Viola, F., Rezende, D. J., Eslami, S., and Teh, Y. W. Neural processes. *arXiv preprint arXiv:1807.01622*, 2018b.

Ghosal, S. and Van der Vaart, A. *Fundamentals of nonparametric Bayesian inference*, volume 44. Cambridge University Press, 2017.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., et al. Array programming with numpy. *Nature*, 585(7825):357–362, 2020.

Hendrycks, D. and Gimpel, K. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Sy2fzU9gl.

Hunter, J. D. Matplotlib: A 2D graphics environment. *IEEE Annals of the History of Computing*, 9(03):90–95, 2007.

Kim, H., Mnih, A., Schwarz, J., Garnelo, M., Eslami, A., Rosenbaum, D., Vinyals, O., and Teh, Y. W. Attentive neural processes. In *International Conference on Learning Representations*, 2018.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kingma, D. P. and Welling, M. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

Kissas, G., Seidman, J. H., Guilhoto, L. F., Preciado, V. M., Pappas, G. J., and Perdikaris, P. Learning operators with coupled attention. *Journal of Machine Learning Research*, 23(215):1–63, 2022.

Kobeissi, H. and Lejeune, E. Mechanical mnist-cahn-hilliard. 2022.

Kosiorek, A. R., Strathmann, H., Zoran, D., Moreno, P., Schneider, R., Mokrá, S., and Rezende, D. J. Nerf-vae: A geometry aware 3d scene generative model. In *International Conference on Machine Learning*, pp. 5742–5752. PMLR, 2021.

Kovachki, N., Lanthaler, S., and Mishra, S. On universal approximation and error bounds for fourier neural operators. *Journal of Machine Learning Research*, 22:Art–No, 2021.

Kuo, H.-H. Gaussian measures in banach spaces. In *Gaussian Measures in Banach Spaces*, pp. 1–109. Springer, 1975.

Lanthaler, S., Mishra, S., and Karniadakis, G. E. Error estimates for DeepOnets: A deep learning framework in infinite dimensions. *Transactions of Mathematics and Its Applications*, 6(1):tnac001, 2022.

Lasanen, S., Roininen, L., and Huttunen, J. M. Elliptic boundary value problems with Gaussian white noise loads. *Stochastic Processes and their Applications*, 128 (11):3607–3627, 2018.

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020.

Lu, L., Jin, P., Pang, G., Zhang, Z., and Karniadakis, G. E. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

Muandet, K., Fukumizu, K., Sriperumbudur, B., Schölkopf, B., et al. Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends® in Machine Learning*, 10(1-2):1–141, 2017.

Müller, T., Evans, A., Schied, C., and Keller, A. Instant neural graphics primitives with a multiresolution hash encoding. *arXiv preprint arXiv:2201.05989*, 2022.

Nadler, B., Lafon, S., Coifman, R. R., and Kevrekidis, I. G. Diffusion maps, spectral clustering and reaction coordinates of dynamical systems. *Applied and Computational Harmonic Analysis*, 21(1):113–127, 2006.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037, 2019.

Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., and Courville, A. On the spectral bias of neural networks. In *International Conference on Machine Learning*, pp. 5301–5310. PMLR, 2019.

Rahman, M. A., Florez, M. A., Anandkumar, A., Ross, Z. E., and Azizzadenesheli, K. Generative adversarial neural operators. *Transactions on Machine Learning Research*, 2022a. URL https://openreview.net/forum?id=X1VzbBU6xZ.

Rahman, M. A., Ross, Z. E., and Azizzadenesheli, K. U-no: U-shaped neural operators. *arXiv preprint arXiv:2204.11127*, 2022b.

Rebain, D., Matthews, M. J., Yi, K. M., Sharma, G., Lagun, D., and Tagliasacchi, A. Attention beats concatenation for conditioning neural fields. *arXiv preprint arXiv:2209.10684*, 2022.

Rosen, P. A., Gurrola, E., Sacco, G. F., and Zebker, H. The InSAR scientific computing environment. In *EUSAR 2012; 9th European conference on synthetic aperture radar*, pp. 730–733. VDE, 2012.

Santhanam, V., Morariu, V. I., and Davis, L. S. Generalized deep image to image regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5609–5619, 2017.

Seidman, J. H., Kissas, G., Perdikaris, P., and Pappas, G. J. NOMAD: Nonlinear manifold decoders for operator learning. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=5OWV-sZvMl.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Skorokhodov, I., Ignatyev, S., and Elhoseiny, M. Adversarial generation of continuous images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10753–10764, 2021.

Sohn, K., Lee, H., and Yan, X. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28, 2015.

Song, J. and Li, B. Nonlinear and additive principal component analysis for functional data. *Journal of Multivariate Analysis*, 181:104675, 2021.

Sriperumbudur, B. K., Fukumizu, K., and Lanckriet, G. R. Universality, characteristic kernels and RKHS embedding of measures. *Journal of Machine Learning Research*, 12 (7), 2011.

Stroock, D. W. *Probability theory: an analytic view*. Cambridge university press, 2010.

Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., and Ng, R. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems*, 33: 7537–7547, 2020.

Vahdat, A. and Kautz, J. Nvae: A deep hierarchical variational autoencoder. *Advances in Neural Information Processing Systems*, 33:19667–19679, 2020.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

Wang, J.-L., Chiou, J.-M., and Müller, H.-G. Functional data analysis. *Annual Review of Statistics and Its Application*, 3:257–295, 2016.

Wang, S., Wang, H., and Perdikaris, P. Improved architectures and training algorithms for deep operator networks. *Journal of Scientific Computing*, 92(2):35, 2022a.

Wang, S., Wang, H., Seidman, J. H., and Perdikaris, P. Random weight factorization improves the training of continuous neural representations. *arXiv preprint arXiv:2210.01274*, 2022b.

Wen, G., Li, Z., Azizzadenesheli, K., Anandkumar, A., and Benson, S. M. U-fno—an enhanced Fourier neural operator-based deep-learning model for multiphase flow. *Advances in Water Resources*, 163:104180, 2022.

Wild, V. and Wynne, G. Variational gaussian processes: A functional analysis view. *arXiv preprint arXiv:2110.12798*, 2021.

Wild, V. D., Hu, R., and Sejdinovic, D. Generalized variational inference in function spaces: Gaussian measures meet bayesian deep learning. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=mMT8bhVBoUa.

Wynne, G. and Duncan, A. B. A kernel two-sample test for functional data. *Journal of Machine Learning Research*, 23(73):1–51, 2022.

Xie, Y., Takikawa, T., Saito, S., Litany, O., Yan, S., Khan, N., Tombari, F., Tompkin, J., Sitzmann, V., and Sridhar, S. Neural fields in visual computing and beyond. In *Computer Graphics Forum*, volume 41, pp. 641–676. Wiley Online Library, 2022.

A. Review of Gaussian Measures

Here we review some basic facts on Gaussian Measures defined Banach spaces. Our presentation follows the discussion in (Kuo, 1975; Bogachev, 2015) and Chapter 8 of (Stroock, 2010). We call a probability measure $\mathbb{P}$ on the Borel σ-algebra $\mathcal{B}(\mathcal{X})$ a *Gaussian measure* if for every $f \in \mathcal{X}^*$, the pushforward measure $\mathbb{P} \circ f^{-1}$ is Gaussian on $\mathbb{R}$. For such a measure there exists two objects that completely characterize it. The *mean* is an element $m \in \mathcal{X}$ such that for all $f \in \mathcal{X}^*$,

$$(f, m) = \int_{\mathcal{X}} (f, x) \, \mathrm{d}\mathbb{P}(x), \tag{12}$$

and the covariance operator $\mathcal{C} : \mathcal{X}^* \to \mathcal{X}$ is defined by

$$\mathcal{C}f(g) = \int_{\mathcal{X}} (f(x) - (f, m))(g(x) - (g, m)) \, \mathrm{d}\mathbb{P}(x). \tag{13}$$

We see from the definition that the covariance operator can also be thought of as a bi-linear form on $\mathcal{X}^*$.

The definition of a Gaussian measure given above allows us to view each $f \in \mathcal{X}^*$ as an element of

$$L^2(\mathbb{P}) := \left\{ f : \mathcal{X} \to \mathbb{R} \;\middle|\; \int_{\mathcal{X}} f(x)^2 \, \mathrm{d}\mathbb{P}(x) < \infty \right\}.$$

Thus, we have an embedding of $\mathcal{X}^* \to L^2(\mathbb{P})$. Forming the completion of the image of this embedding with respect to the $L^2(\mathbb{P})$ norm forms the *Cameron Martin* space of the measure $\mathbb{P}$, denoted $\mathcal{H}_{\mathbb{P}}$. Note that by construction, $\mathcal{X}^*$ can be identified with a dense subspace of $\mathcal{H}_{\mathbb{P}}$ and we have the inclusion map $\mathcal{I} : \mathcal{H}_{\mathbb{P}} \to L^2(\mathbb{P})$. The map $\mathcal{I}$ is sometimes known as the Paley-Wiener map (for example, when $\mathbb{P}$ is the Wiener measure $\mathcal{I}(h)(x)$ is the Ito integral $\int \dot{h} \, \mathrm{d}x(t)$). Note that this implies for any $h \in \mathcal{H}_{\mathbb{P}}$, the quantity

$$\langle h, x \rangle^{\sim} := \mathcal{I}(h)(x)$$

is well defined for $\mathbb{P}$-almost-every x. It can additionally be shown that there exists a dense injection $\mathcal{H}_{\mathbb{P}} \hookrightarrow X$. The Cameron Martin space has a number of equivalent definitions and determines many of the relevant properties of the measure $\mathbb{P}$.

We will make repeated use of the *Cameron Martin Theorem*, which determines when the translation of a Gaussian measure $\mathbb{P}$ by $h \in X$ gives an equivalent measure. This will allow us to give an expression for the log likelihood as well as the KL divergence term in the ELBO objective. For a proof see (Bogachev, 2015), (Kuo, 1975) or (Stroock, 2010).

Theorem A.1 (Cameron-Martin)**.** *Given a Gaussian measure $\mathbb{P}$ on $\mathcal{X}$ with Cameron Martin space $\mathcal{H}_{\mathbb{P}}$, the translated measure $\mathbb{P}_h(A) := \mathbb{P}(A - h)$ is absolutely continuous with respect to $\mathbb{P}$ if and only if $h \in \mathcal{H}_{\mathbb{P}}$, with*

$$\log \frac{\mathrm{d}\mathbb{P}_h}{\mathrm{d}\mathbb{P}}(x) = -\frac{1}{2}\|h\|^2_{\mathcal{H}_{\mathbb{P}}} + \langle h, x \rangle^{\sim}. \tag{14}$$

When the Gaussian measure $\mathbb{P}$ is supported on a Hilbert space $\mathcal{H}$, the previous definitions simplify due to the Riesz representation theorem which allows us to use the isomorphism $\mathcal{H}^* \simeq \mathcal{H}$. In particular, for this case the covariance operator is a trace-class, self-adjoint operator $\mathcal{C} : \mathcal{H} \to \mathcal{H}$. Further, in this case the Cameron Martin space can be identified with $\mathrm{im}(\mathcal{C}^{1/2})$ and has an inner product given by

$$\langle x, y \rangle_{\mathcal{H}_{\mathbb{P}}} = \langle \mathcal{C}^{-1/2} x, \mathcal{C}^{-1/2} y \rangle_{\mathcal{H}}.$$

A.1. Abstract Wiener Space

There is an alternate characterization of Gaussian measures which instead begins with a separable infinite dimensional Hilbert space $\mathcal{H}$. If we attempt to sample from the "standard Gaussian" on $\mathcal{H}$ by taking an orthonormal basis $\{e_i\}_{i=1}^{\infty}$ and try to form the sum

$$x = \sum_{i=1}^{\infty} \xi_i e_i, \quad \xi_i \sim \mathcal{N}(0, 1), \text{ i.i.d.}, \tag{15}$$

we see that $\mathbb{E}[\|x\|^2_{\mathcal{H}}] = \infty$ and thus $x \notin \mathcal{H}$ almost surely.

The way around this is to consider the convergence of the sum (15) with respect to a norm other than that from $\mathcal{H}$. After picking such a "measurable norm" (Kuo, 1975) we may complete the space $\mathcal{H}$ with respect to this norm to obtain a new Banach space X, on which the measure we have tried to construct is supported. This completion gives us a dense inclusion $i : \mathcal{H} \hookrightarrow X$. The triple $(i, \mathcal{H}, X)$ is called an *Abstract Wiener Space* (AWS). The induced Gaussian measure $\mathbb{P}$ on X has $\mathcal{H}$ as its Cameron Martin space. Thus, the AWS construction gives us a way to construct a Gaussian measure starting from its Cameron Martin space.

An example of this construction that will be particularly useful is that which starts from $L^2(X;\mathbb{R})$ with $X \subset \mathbb{R}^d$ compact as the desired Cameron Martin space. If we consider the dense inclusion $L^2(X;\mathbb{R}) \hookrightarrow \mathcal{H}^{-s}(X;\mathbb{R})$, with $s > d/2$, then $(i, L^2(X;\mathbb{R}), \mathcal{H}^{-s}(X;\mathbb{R}))$ is an AWS and the associated measure is called the *white noise measure*.

B. Proof of Theorem 4.1

To prove the generalization of the ELBO, we will need to use a modified measure-theoretic formulation of Bayes theorem phrased in terms of Radon-Nikodym derivatives. We begin from that presented in (Ghosal & Van der Vaart, 2017),

$$\frac{\mathrm{d}\mathbb{P}_{z|u}}{\mathrm{d}\mathbb{P}_z} = \frac{1}{c(u)} \frac{\mathrm{d}\mathbb{P}_{u|z}}{\mathrm{d}\mathbb{P}_u}, \tag{16}$$

where

$$c(u) = \int_{\mathcal{Z}} \frac{\mathrm{d}\mathbb{P}_{u|z}}{\mathrm{d}\mathbb{P}_u}(u)\ \mathrm{d}\mathbb{P}_z. \tag{17}$$

We claim that $c(u) = 1$, $\mathbb{P}_u$-a.e. Since the Radon-Nikodym derivative is non-negative, we have that $c(u) \geq 0$. Next, we show that $c(u) \leq 1$. Assume this is not true. Then by the disintegration property of the regular conditional measures we may write for any measurable $f(u)$,

$$\begin{aligned}
\int_{\mathcal{X}} f(u)\ \mathrm{d}\mathbb{P}_u &= \int_{\mathcal{Z}} \int_{\mathcal{X}} f(x)\ \mathrm{d}\mathbb{P}_{u|z}\ \mathrm{d}\mathbb{P}_z \\
&= \int_{\mathcal{X}} f(u) \left(\int_{\mathcal{Z}} \frac{\mathrm{d}\mathbb{P}_{u|z}}{\mathrm{d}\mathbb{P}_u}\ \mathrm{d}\mathbb{P}_z \right) d\mathbb{P}_u \\
&> \int_{\mathcal{X}} f(u)\ \mathrm{d}\mathbb{P}_u,
\end{aligned}$$

which is a contradiction. Hence, $c(u) \leq 1$. This allows us to write

$$\begin{aligned}
\int_{\mathcal{X}} |1 - c(x)|\ \ \mathrm{d}\mathbb{P}_u &= \int_{\mathcal{Z}} 1 - c(x)\ \ \mathrm{d}\mathbb{P}_u \\
&= \int_{\mathcal{X}} 1 - \int_{\mathcal{Z}} \frac{\mathrm{d}\mathbb{P}_{u|z}}{\mathrm{d}\mathbb{P}_u}(u)\ \mathrm{d}\mathbb{P}_z\ \mathrm{d}\mathbb{P}_u \\
&= 1 - \int_{\mathcal{Z}} \int_{\mathcal{X}} \frac{\mathrm{d}\mathbb{P}_{u|z}}{\mathrm{d}\mathbb{P}_u}(u)\ \mathrm{d}\mathbb{P}_u\ \mathrm{d}\mathbb{P}_z \\
&= 1 - \int_{\mathcal{Z}} \int_{\mathcal{X}} \ \mathrm{d}\mathbb{P}_u\ \mathrm{d}\mathbb{P}_x \\
&= 0,
\end{aligned}$$

and we have shown that $c(x) = 1$ $\mathbb{P}_u$-a.e and thus from (16)

$$\frac{\mathrm{d}\mathbb{P}_{z|u}}{\mathrm{d}\mathbb{P}_z} = \frac{\mathrm{d}\mathbb{P}_{u|z}}{\mathrm{d}\mathbb{P}_u}. \tag{18}$$

We are now able to prove the generalize ELBO stated in Theorem 4.1. By the definition of the KL divergence,

$$\mathrm{KL}[\mathbb{Q}^{\phi}_{z|u} || \mathbb{P}_{z|u}] = \int_{\mathcal{Z}} \log \left(\frac{\mathrm{d}\mathbb{Q}^{\phi}_{z|u}}{\mathrm{d}\mathbb{P}_{z|u}}(z) \right)\ \ \mathrm{d}\mathbb{Q}^{\phi}_{z|u}. \tag{19}$$

From the chain rule for Radon-Nikodym derivatives we may write

$$\frac{\mathrm{d}\mathbb{Q}^{\phi}_{z|u}}{\mathrm{d}\mathbb{P}_{z|u}} = \frac{\mathrm{d}\mathbb{Q}^{\phi}_{z|u}}{\mathrm{d}\mathbb{P}_{z}} \frac{\mathrm{d}\mathbb{P}_{z}}{\mathrm{d}\mathbb{P}_{z|u}}. \tag{20}$$

Using (18), we then have

$$\begin{aligned}
\frac{\mathrm{d}\mathbb{Q}^{\phi}_{z|u}}{\mathrm{d}\mathbb{P}_{z|u}} &= \frac{\mathrm{d}\mathbb{Q}^{\phi}_{z|u}}{\mathrm{d}\mathbb{P}_{z}} \frac{\mathrm{d}\mathbb{P}_{u}}{\mathrm{d}\mathbb{P}_{u|z}} \\
&= \frac{\mathrm{d}\mathbb{Q}^{\phi}_{z|u}}{\mathrm{d}\mathbb{P}_{z}} \frac{\mathrm{d}\mathbb{P}_{u}}{\mathrm{d}\mathbb{V}} \frac{\mathrm{d}\mathbb{V}}{\mathrm{d}\mathbb{P}_{u|z}} \\
&= \frac{\mathrm{d}\mathbb{Q}^{\phi}_{z|u}}{\mathrm{d}\mathbb{P}_{z}} \frac{\mathrm{d}\mathbb{P}_{u}}{\mathrm{d}\mathbb{V}} \left(\frac{\mathrm{d}\mathbb{P}_{u|z}}{\mathrm{d}\mathbb{V}} \right)^{-1},
\end{aligned} \tag{21}$$

where the last equality holds under the assumption of mutual absolute continuity between all written measures. Placing this relation in (19) shows that

$$\begin{aligned}
\mathrm{KL}[\mathbb{Q}^{\phi}_{z|u} || \mathbb{P}_{z|u}] &= \int_{\mathcal{Z}} \log \left(\frac{\mathrm{d}\mathbb{Q}^{\phi}_{z|u}}{\mathrm{d}\mathbb{P}_{z}} \frac{\mathrm{d}\mathbb{P}_{u}}{\mathrm{d}\mathbb{V}} \left(\frac{\mathrm{d}\mathbb{P}_{u|z}}{\mathrm{d}\mathbb{V}} \right)^{-1} \right) \, \mathrm{d}\mathbb{Q}^{\phi}_{z|u} \\
&= \int_{\mathcal{Z}} \log \left(\frac{\mathrm{d}\mathbb{Q}^{\phi}_{z|u}}{\mathrm{d}\mathbb{P}_{z}} \right) \, \mathrm{d}\mathbb{Q}^{\phi}_{z|u} + \int_{\mathcal{Z}} \log \left(\frac{\mathrm{d}\mathbb{P}_{u}}{\mathrm{d}\mathbb{V}} \right) \, \mathrm{d}\mathbb{Q}^{\phi}_{z|u} - \int_{\mathcal{Z}} \log \left(\frac{\mathrm{d}\mathbb{P}_{u|z}}{\mathrm{d}\mathbb{V}} \right) \, \mathrm{d}\mathbb{Q}^{\phi}_{z|u}.
\end{aligned} \tag{22}$$

We identify the first term on the right as $\mathrm{KL}[\mathbb{Q}^{\phi}_{z|u} || \mathbb{P}_{z}]$. The integrand of the second term is a function of u only and $\mathbb{Q}^{\phi}_{z|u}$ is a probability measure, thus the second term is equal to the log likelihood of the data x. The third term is the expectation of the conditional likelihood of x given z. We have thus proved the equality

$$\mathrm{KL}[\mathbb{Q}^{\phi}_{z|u} || \mathbb{P}_{z|u}] = \mathrm{KL}[\mathbb{Q}^{\phi}_{z|u} || \mathbb{P}_{z}] + \log \left(\frac{\mathrm{d}\mathbb{P}_{u}}{\mathrm{d}\mathbb{V}}(x) \right) - \mathbb{E}_{z \sim \mathbb{Q}^{\phi}_{z|u}} \left[\log \frac{\mathrm{d}\mathbb{P}_{u|z}}{\mathrm{d}\mathbb{V}(x)} \right]. \tag{23}$$

C. Architectures

Here we present different architecture choices we have considered in the different experiments presented in this manuscript. Specifically here we outline our specific choices in terms of encoder and decoder architectures, as well as different types of positional encodings.

C.1. Encoders

We consider two types of encoders in our benchmarks. In the Gaussian Random Field we consider a Multi-layer Perceptron (MLP) network encoder. In all other benchmarks we build encoders using a simple VGG-style deep convolutional network (Simonyan & Zisserman, 2014), where in each layer the input feature maps are down-sampled by a factor of 2 using strided convolutions, while the number of channels are doubled.

C.2. Decoders

First, we consider possible decoder choices. The decoders can be categorized broadly as linear and nonlinear.

Linear Decoder: Linear decoders take the form

$$\mathcal{D}(z)(x) = \sum_{i=1}^{n} z_i \tau_i(\gamma(x)), \quad z_i \sim \mathcal{N}(0,1), \quad \text{i.i.d.},$$

where τ is a vector-valued function parameterized by a Multi-layer Perceptron Network (MLP), and $\gamma(x)$ is a positional encoding of the query locations (see next section for more details).

Nonlinear Decoders: Generalized nonlinear decoders can be constructed as

$$\mathcal{D}(z)(x) = f_\theta(z_i, \gamma(x)), \quad z_i \sim \mathcal{N}(0,1), \quad \text{i.i.d.},$$

where f is a function parameterized by an MLP network, and $\gamma(x)$ is a positional encoding of the query locations (see next section for more details). Following the work of Rebain et. al. (Rebain et al., 2022) we consider two types of conditioning f on the latent vector z. The first approach concatenates the latent vector z with the query location $\gamma(x)$ before pushing them through the decoder, as in (Seidman et al., 2022). An alternative approach is to split the latent vector into chunks and concatenate each chunk to each hidden layer of the decoder, see (Rebain et al., 2022) for more details.

C.3. Positional Encodings

Positional encodings have been shown to help coordinate-MLPs to capture higher frequency components of signals thereby mitigating spectral bias (Mildenhall et al., 2021; Tancik et al., 2020;?). Here we employ different types of positional encodings depending on the nature of the benchmark we are considering.

Fourier Features: First we consider a periodic encoding of the form

$$\gamma(x) = [1, \cos(\omega x), \sin(\omega x), \ldots, \cos(k\omega x), \sin(k\omega x)],$$

with $\omega = \frac{2\pi}{L}$, and some non-negative integer k. Here L denotes the length of the domain. This encoding was employed in the GRF benchmark to ensure that the generated functions satisfy a zero Dirichlet boundary condition at the domain boundaries.

Random Fourier Features: The Random Fourier Feature encoding can be written as:

$$\gamma(x) = [\cos(2\pi B x), \sin(2\pi B x)],$$

where $B \in \mathbb{R}^{q \times d}$ is sampled from the Gaussian distribution $\mathcal{N}(0, \sigma^2)$ using a user-specified variance parameter σ. Here q denotes the number of Fourier features used and d is the dimension of the query point x. In our experiments we set $q = n/2$, where n is the latent space dimension of the model. We empirically found that this positional encoding gives good performance for the Cahn-Hilliard data-set.

Multi-resolution Hash Encoding: For cases where we expect the function to contain multiscale characteristics, such as in the InSAR data-set, we considered the Multi-resolution Hash Encoding proposed in (Müller et al., 2022). We found that this type of encoding gives the best performance for the InSar data-set.

C.4. Hyper-parameter Sweeps

In order to quantify the sensitivity of our results on the choice of different hyper-parameters, as well as to identify hyper-parameter settings that lead to good performance, we have performed a series of hyper-parameter sweeps for each benchmark considered in the main manuscript. Here we present the sweep settings and the best configuration identified for each benchmark.

Gaussian Random Field: We perform a parametric sweep by considering different latent dimension sizes and different model initialization seeds, as shown in Table 3. Our goal is to find the model out of these set-ups that minimizes the normalized Hilbert-Schmidt norm between samples drawn from the ground truth data distribution and the VANO models. The lowest Hilbert-Schmidt norm value is achieved for $n = 64$.

2D Gaussian Densities: We perform a parametric sweep by considering different latent dimension sizes and different model initialization seeds, as shown in Table 3. Our goal is to find the model out of these set-ups that minimizes the generalized MMD metric between samples drawn from the ground truth data distribution and the VANO models. The lowest generalized MMD value is achieved for $n = 32$.

Cahn-Hilliard: We perform a parametric sweep by considering different latent dimension sizes, KL loss weights β and decoder types, as shown in Table 3. Our goal is to find the model out of these set-ups that minimize the generalized MMD metric between samples drawn from the ground truth data distribution and the VANO models. The lowest generalized MMD value is achieved for $n = 64$, $\beta = 10^{-4}$ and a concatenation decoder.

Table 3. Hyper-parameter sweep settings for different examples (C and SC indicate a concatenation and a split concatenation decoder, respectively, see Section C.2).

BENCHMARK	RANDOM SEED	n	β	DECODER	LAYER WIDTH
GRF	[0, . . . , 10]	[2,4,8,16,32,64]	-	-	-
2D GAUSSIAN DENSITIES	[0, . . . , 4]	[4, 32, 64, 128, 256]	-	-	-
CAHN-HILLIARD	-	[32, 64, 128]	$[10^{-5}, 10^{-4}, 10^{-3}]$	[C,SC]	[64, 128, 256]
INSAR INTERFEROGRAMS	-	[128, 256, 512]	$[10^{-5}, 10^{-4}, 10^{-3}]$	-	[256, 512]

InSAR Interferograms: We perform a parametric sweep by considering different latent dimension sizes, KL loss weight β and decoder layer width, as shown in Table 3. Our goal is to find the model out of these set-ups that minimize the generalized MMD metric between samples drawn from the ground truth data distribution and the VANO models. The lowest generalized MMD value is achieved for $n = 256$, $\beta = 10^{-4}$ and a decoder layer width of 512.

D. Experimental Details

In this section we present details about each experimental set-up and the generation of the training and testing data-sets.

D.1. Gaussian Random Field

Data Generation: From the Karhunen-Loéve theorem (Adler, 1990) we can construct random functions distributed according to a mean zero Gaussian measure with covariance Γ

$$u = \sum_{i=1}^{\infty} \xi_i \sqrt{\lambda_i} \phi_i, \tag{24}$$

where $\xi_i \sim \mathcal{N}(0, 1)$ are i.i.d. normally distributed random variables, and λ_i and ϕ_i are the eigenvalues and eigenvectors of the covariance operator.

We define $\lambda_i = ((2\pi i)^2 + \tau^2)^{-\alpha}$ and $\phi_i(x) = \sqrt{2}\sin(2\pi i x)$. These setup corresponds to the eigenvalues and eigenfunctions of the operator $(I + \Delta)^{-\alpha}$ on $X = [0, 1]$ with zero boundary conditions. To generate these functions, we take samples of the sum in (11) truncated at the first 32 eigenfunctions and eigenvalues to construct $N_{train} = 2048$ functions evaluated at $m = 128$ points which we use to train our model. We independently sample an additional $N_{test} = 2048$ functions for testing.

Encoder: We parameterize the encoder using an MLP network with 3 hidden layers, width of 128 neurons and Gelu (Hendrycks & Gimpel, 2016) activation functions.

Decoder: We consider a linear decoder parameterized by a 3-layer deep MLP network, with a width of 128 neurons width, periodic positional encoding and Gelu activation functions.

Training Details: We consider a latent space dimension of $n = 64$, $S = 16$ Monte Carlo samples for evaluating the expectation in the reconstruction loss, and a KL loss weighting factor $\beta = 5\ 10^{-6}$. We train the model using the Adam optimizer (Kingma & Ba, 2014) with random weight factorization (Wang et al., 2022b) for $40,000$ training iterations with a batch size of 32 and a starting learning rate of 10^{-3} with exponential decay of 0.9 every 1,000 steps.

Additionally, it has been observed in the operator learning literature (Di Leoni et al., 2023; Wang et al., 2022a) that when a functional data-set has examples of varying magnitudes the training can overfit those with larger magnitude, as they are more heavily weighted in the loss function. To correct for this, we use a data dependent scaling factor in the likelihood terms. We find this modification is only necessary for the first two experiments, as the other scenarios do not have this magnitude variability in their data.

Evaluation: To test the performance of the linear decoder VANO model, we train it on a range of dimensions for its latent space, $n \in \{2, 4, 8, 16, 32, 64\}$. We then generate data by sampling latent vectors from the prior $z \sim \mathbb{P}_z = \mathcal{N}(0, \mathrm{I})$

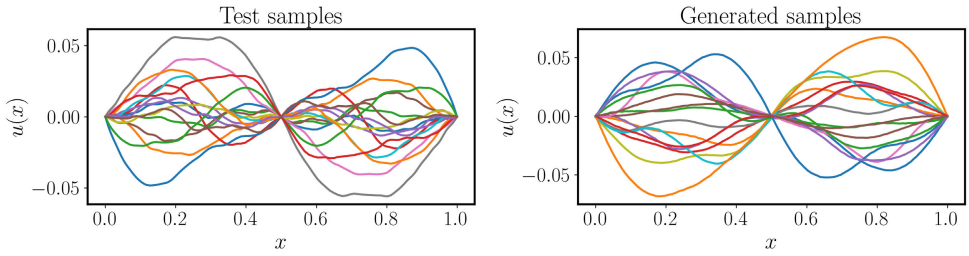

Figure 6. GRF benchmark: Left: Functions sampled from the ground truth datset, Right: Functions sampled from the VANO model.

and map them through the decoder to create functions as in (6.1). Since this distribution is a mean zero Gaussian, like the ground truth Gaussian random field, it suffices to compare the resulting covariance operators to measure how close the two distributions are. The covariance associated with the distribution described in (6.1) is given by

$$\hat{\Gamma} = \sum_{i=1}^{n} \tau_i \otimes \tau_i. \tag{25}$$

We use the ground truth covariance eigenfunctions and the approximated eigenfunctions to compute the true and approximated covariance operators, and compare them using the Hilber-Schmidt norm. We observe in Figure 3 that qualitatively the two sets of look similar indicating the model is recovering the optimal choice of decoder.

D.2. 2D Gaussian Densities

Data Generation: For this example we construct a functional data-set consisting of two-dimensional Gaussian pdf functions in the unit square $X = [0, 1]^2$

$$U(x, y) = (2\pi)^{-1/2} \det(\Sigma)^{-1/2} \exp\Big(-\frac{1}{2}(x - \mu)^\top \Sigma^{-1}(x - \mu)\Big), \tag{26}$$

where $\mu \sim \big(\mu_x, \mu_y\big)$ and $\Sigma = \sigma I$. We consider $\mu_x, \mu_y \sim U(0, 1)$ and $\sigma \sim U(0, 0.1) + 0.01$ and generate $N_{train} = 2,048$ training and $N_{test} = 2048$ testing samples of $m = 2,304$ measurement points on a 48×48 equi-spaced grid.

Encoder: We parameterize the encoder using a 4-layer deep VGG-style convolutional network using a 2×2 convolution kernel, a stride of size two, (8, 16, 32, 64) channels at each layer, and Gelu activation functions.

Decoder: We consider two types of decoders, a linear and a nonlinear. The linear decoder is parameterized using a 3-layer deep MLP network with 128 neurons per layer and Gelu activation functions. For the nonlinear decoder case we consider a 3-layer deep MLP network with 128 neurons per layers, Gelu activation functions and concatenation conditioning. In both cases we also apply a softplus activation on the decoder output to ensure positivity in the predicted function values.

Training Details: We consider a latent space dimension of $n = 32$, $S = 4$ Monte Carlo samples for evaluating the expectation of the reconstruction part of the loss and a KL loss weighting factor $\beta = 10^{-5}$. We train the model using the Adam optimizer (Kingma & Ba, 2014) with random weight factorization (Wang et al., 2022b) for $20,000$ training iterations with a batch size of 32 and a starting learning rate of 10^{-3} with exponential decay of 0.9 every 1,000 steps. For this case, we also re-scale the likelihood terms by the empirical norm of the input function to ensure the terms with larger magnitude do not dominate the optimization procedure, as in (Wang et al., 2022a; Di Leoni et al., 2023).

Evaluation: In figure 7 we present the decay of the PCA eigenvalues computed across function samples. The eigenvalue decay is slow which is the reason that makes the VANO with a linear decoder fail in reconstructing and generating function samples as shown in Figures 9 and 10, respectively. We perform a comparison between samples from the true data-set and samples generated by VANO using the generalized MMD metric computed using 512 function samples.

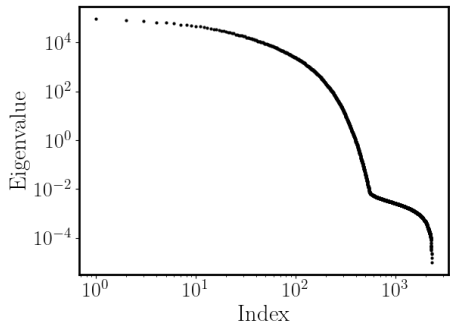

Figure 7. 2D Gaussian densities benchmark: PCA eigenvalue decay of the training data-set.

D.3. Phase Separation Patterns in Cahn-Hilliard Systems

Data Generation: The Cahn-Hilliard equation is a fourth order partial differential equation that describes the evolution of the phase separation process in an anisotropic binary alloys (Kobeissi & Lejeune, 2022). The Cahn-Hilliard equation is written as:

$$\frac{\partial c}{\partial t} - \nabla \cdot (M \nabla (\mu_c - \lambda \nabla^2 c)) = 0 \quad \text{in} \Omega, \tag{27}$$

where $0 < c < 1$ denotes the concentration of one of the components of the binary mixture, M is a mobility parameter, μ_c is the chemical potential of the uniform solution, and λ a positive scalar that describes the thickness of the interfaces of the two mixtures. We consider boundary conditions

$$\begin{aligned} c &= g \quad \text{on } \Gamma_g, \\ M\lambda \nabla c \cdot \boldsymbol{n} &= 0 \quad \text{in } \partial\Omega, \\ M\lambda \nabla c &= r \quad \text{on } \Gamma_r, \\ M\nabla(\mu_c - \lambda\nabla^2 c) \cdot \mathbf{n} &= s \quad \text{on } \Gamma_s, \end{aligned} \tag{28}$$

where Ω is two dimensional domain, $\partial\Omega$ the domain boundary, $\boldsymbol{n}$ the unit outward normal, and $\overline{\Gamma_g \bigcup \Gamma_s} = \overline{\Gamma_q \bigcup \Gamma_r}$, see (Kobeissi & Lejeune, 2022). The initial conditions are given by

$$c(\mathbf{x}, 0) = c_0(\mathbf{x}) \quad \text{in } \Omega. \tag{29}$$

For the above set-up the chemical potential μ_c is chosen as a symmetric double well potential where the wells correspond to the two different material phases, namely $f = bc^2(1 - c^2)$. For more information on the set-up we refer the interested reader to (Kobeissi & Lejeune, 2022).

Encoder: We employ a VGG-style convolutional encoder with 5 layers using 2×2 convolution kernels, stride of size two, (8, 16, 32, 64, 128) channels per layer, and Gelu activation functions.

Decoder: We employ a nonlinear decoder parameterized by a 4-layer deep MLP network with 256 neurons per layer, random Fourier Features positional encoding (Tancik et al., 2020) with $\sigma^2 = 10$, Gelu activation functions and concatenation conditioning. We also apply a sigmoid activation on the decoder outputs to ensure that the predicted function values are in [0,1].

Training Details: We consider a latent space dimension of $n = 64$, $S = 4$ Monte Carlo samples for evaluating the expectation of the reconstruction part of the loss and a KL loss weighting factor $\beta = 10^{-4}$. We train the model using the Adam optimizer (Kingma & Ba, 2014) with random weight factorization (Wang et al., 2022b) for $20,000$ training iterations with a batch size of 16 and a starting learning rate of 10^{-3} with exponential decay of 0.9 every 1,000 steps.

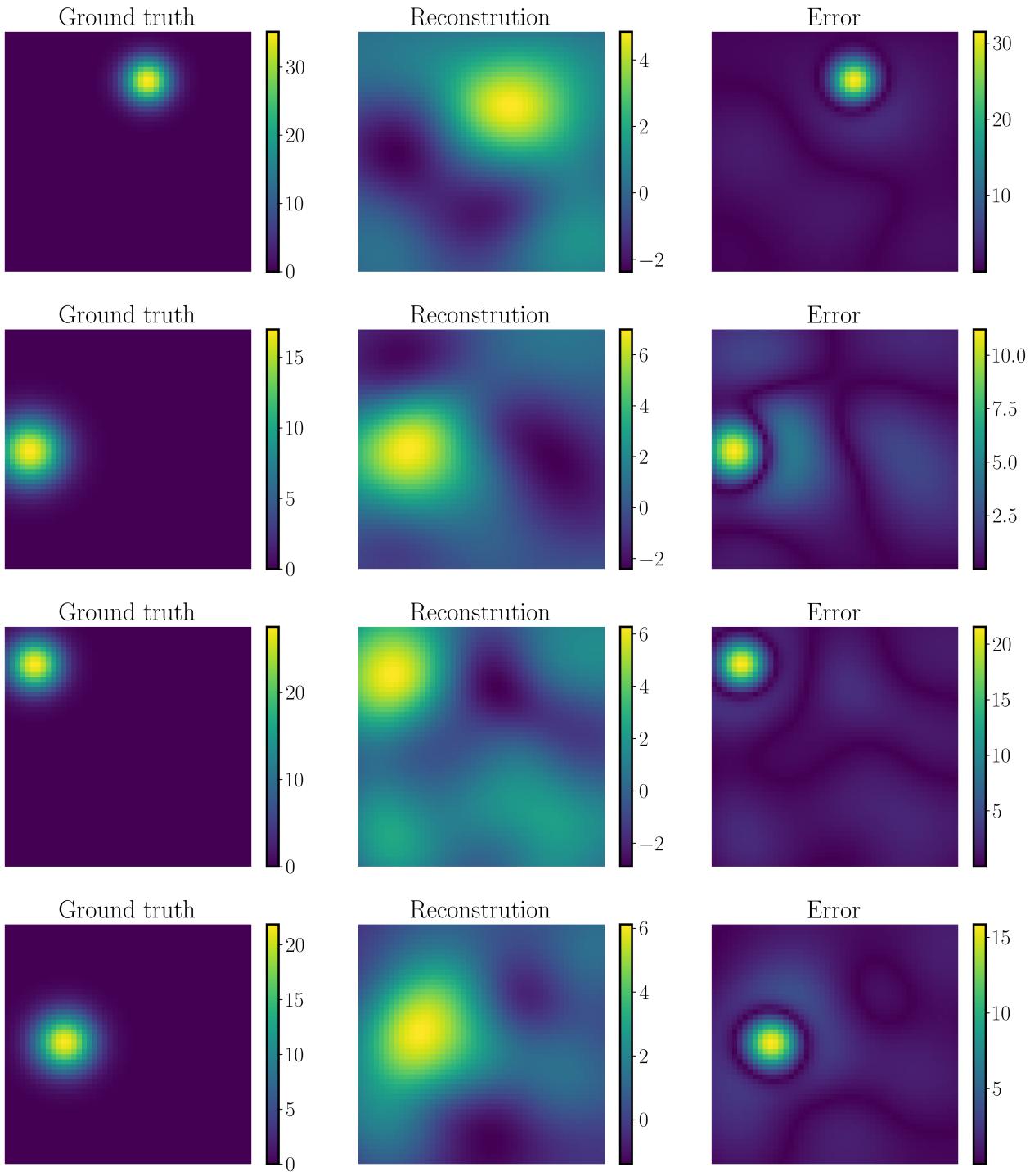

Figure 8. 2D Gaussian densities benchmark: Left: Ground truth functions samples from the test data-set, Middle: Linear VANO reconstruction, Right: Absolute error.

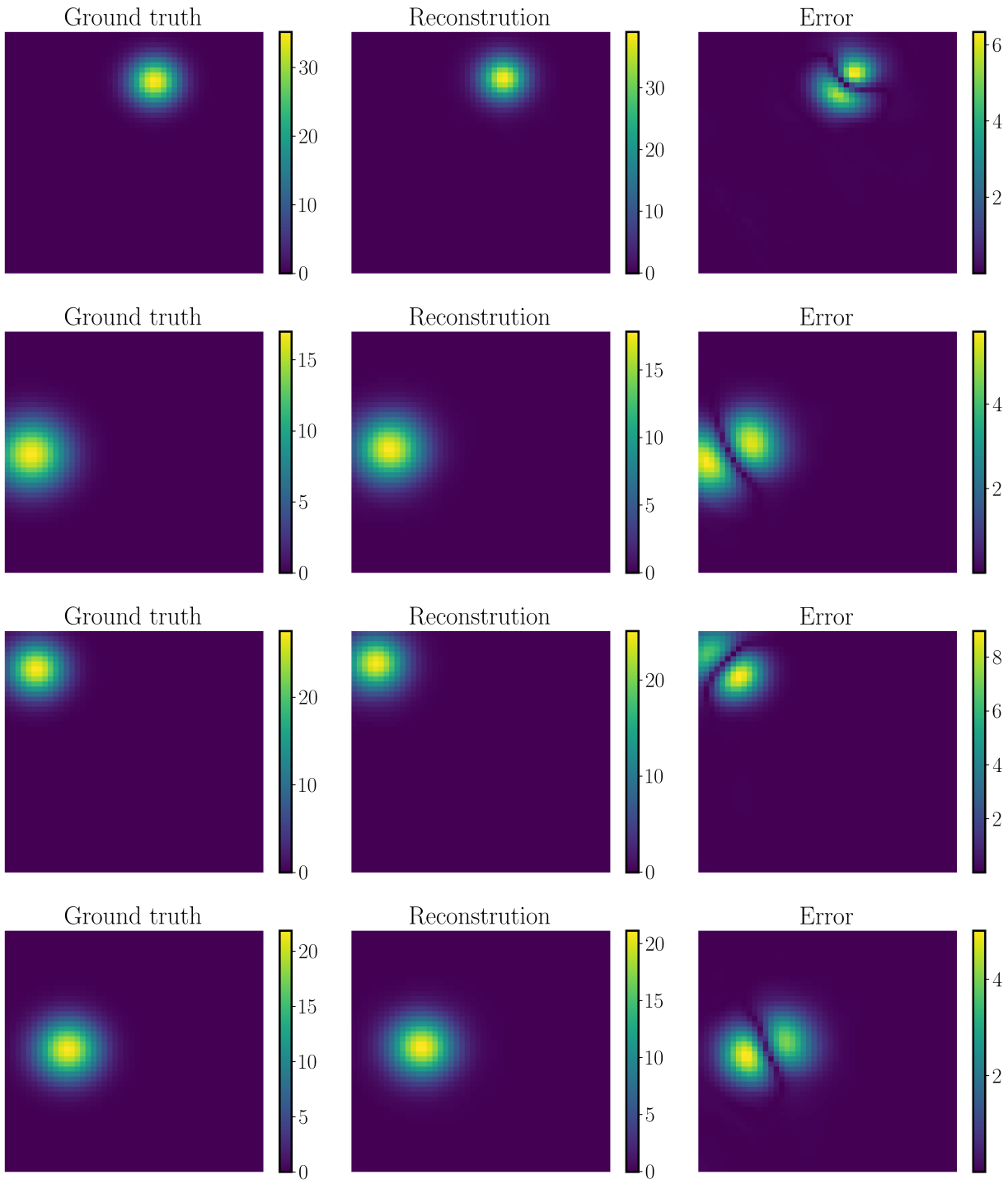

Figure 9. 2D Gaussian densities benchmark: Left: Ground truth functions samples from the test data-set, Middle: Nonlinear VANO reconstruction, Right: Absolute point-wise error.

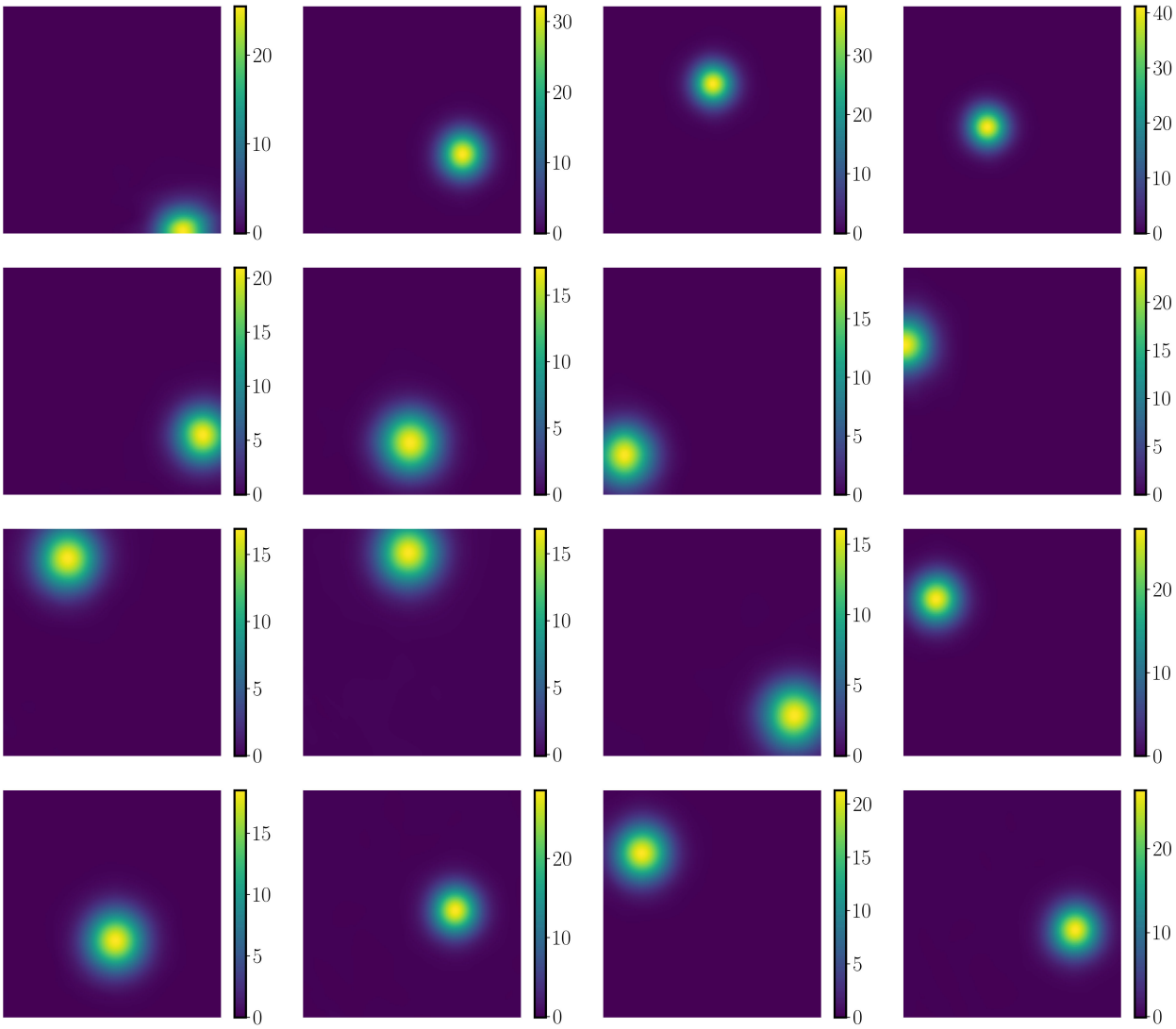

Figure 10. 2D Gaussian densities benchmark: Gaussian density function samples generated from the VANO model with a nonlinear decoder in super-resolution mode (training resolution: 48×48, sample resolution 256×256).

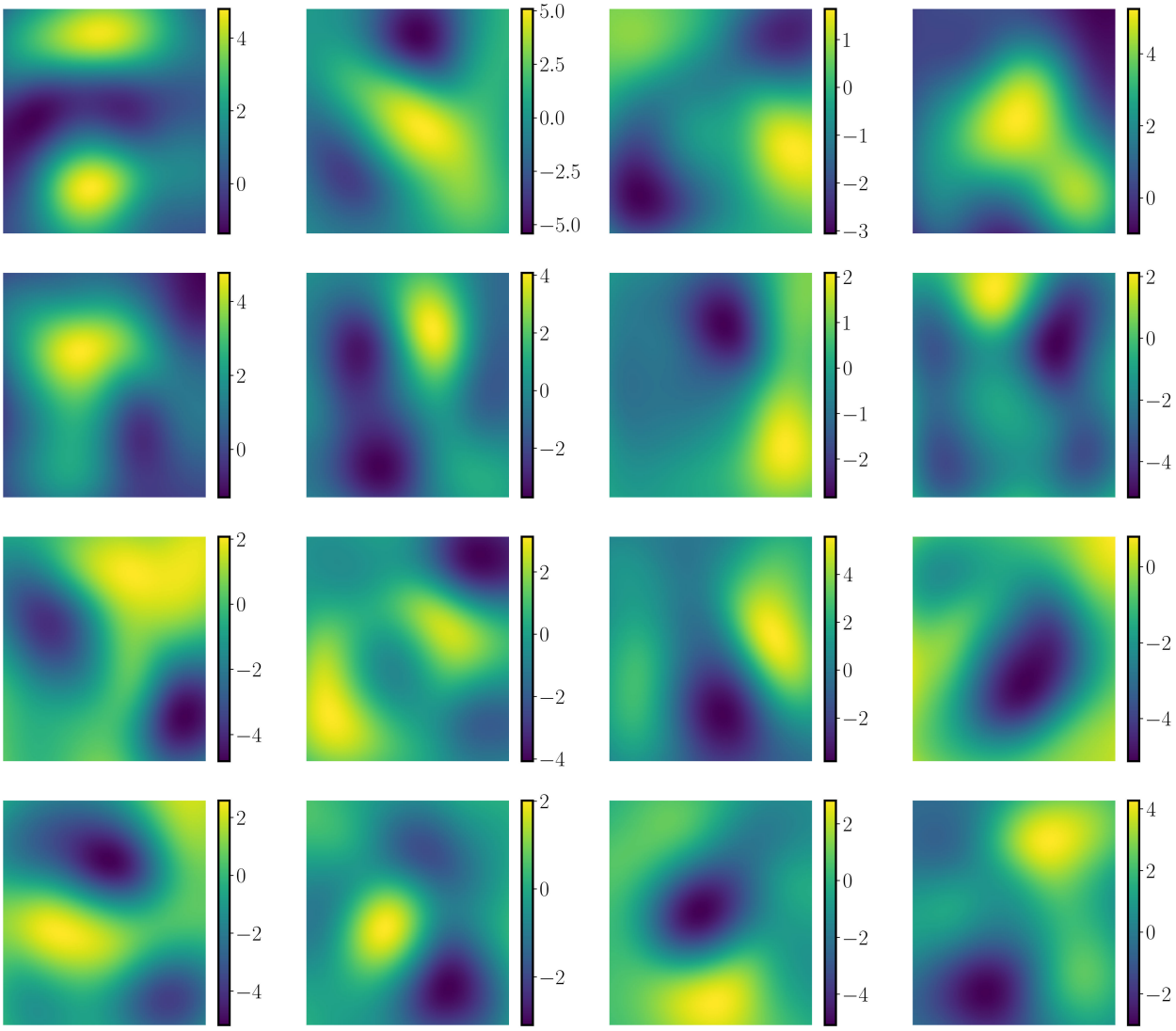

Figure 11. 2D Gaussian densities benchmark: 2D Gaussian density function samples generated from the VANO model using a linear decoder in super-resolution mode (training resolution: 48×48, sample resolution 256×256).

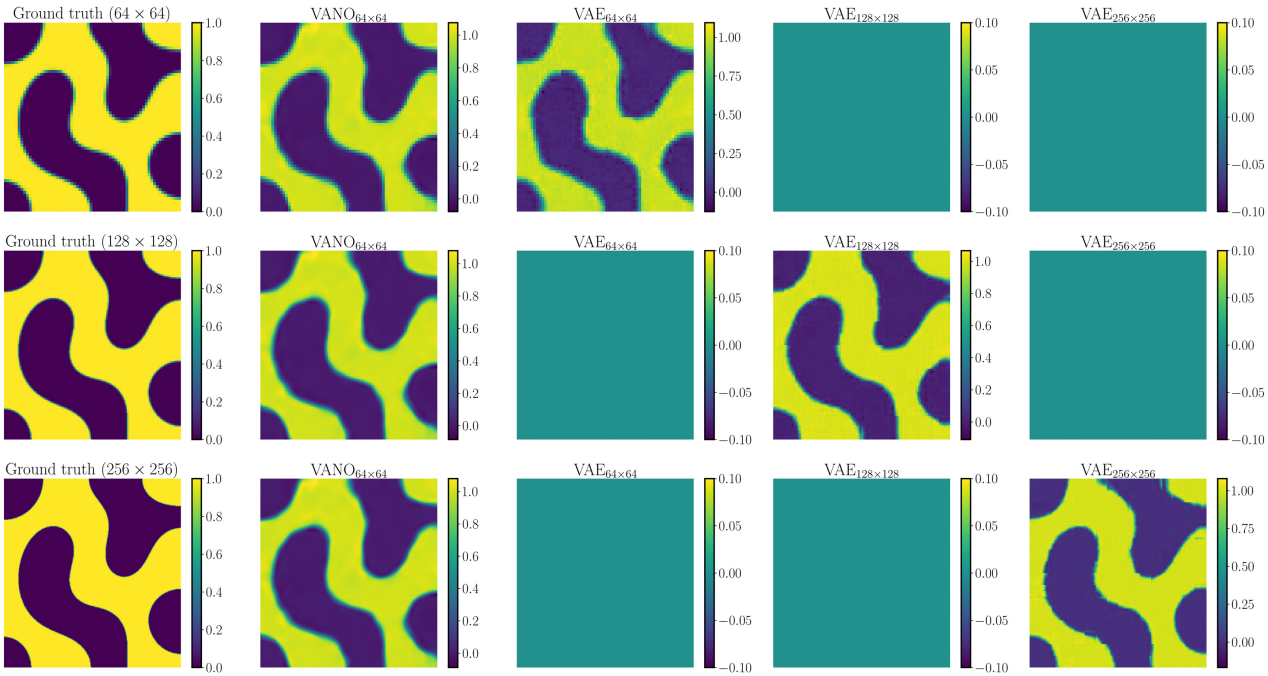

Figure 12. Cahn-Hilliard benchmark: Ground truth data-set samples, VANO reconstructions, and discretize-first VAE reconstructions of Cahn-Hilliard functions in different resolutions. For the VANO model, we train on 64x64 resolution and then perform super-resolution for 128x128 and 256x256, while for the discrete-first VAE models we must train a separate model at each target resolution.

Discretize-first VAE Training Set-up: For the discretize-first VAE simulations we consider an identical encoder as in the VANO case, and a convolutional decoder that exactly mirrors the encoder structure, albeit using transposed convolutions.

Evaluation: We perform a comparison between the ground truth test data-set and samples generated by VANO and the discretize-first VAE models using the generalized MMD metric computed using 256 function samples. We present reconstructed function samples chosen randomly from the test data-set for the VANO model trained on a 64x64 resolution and use to make predictions on higher resolutions, namely 128x128 and 256x256. Separate discretize-first VAE models are trained on data with resolutions 64x64, 128x128 and 256x256. Representative test reconstructions from all models are presented in Figure 12. In Figure 13 we show generated function samples at different resolutions. For the VANO model we train the model on 64x64 resolution images and generate samples in super-resolution mode, while for the discretize-VAE models we present samples of the same resolution as the images on which each model was trained on. The resolution of the images used for training each model are indicated by subscripts.

D.4. Interferometric Synthetic Aperture Radar data-set

Data Generation: InSAR is a sensing technology that exploits radar signals from aerial vehicles in order to record changes in echoes over time to measure the deformation of a point on the Earth between each pass of the aerial vehicle over the specified point. This technology is employed in measuring deformation on the Earth's surface caused by earthquakes, volcanic eruptions, etc. The data returned by InSAR mostly consists of interferograms. An interferogram is an angular-valued spatial field $u \in \mathcal{X}$ and $u(x) \in [-\pi, \pi]$ where $x \in X$ the domain of u which in this case corresponds to the Earth surface. Interferograms contain different types of noise and are affected by local weather, Earth topography, and different passes of the aerial vehicles which makes them complex to approximate (Rahman et al., 2022a).

The InSAR data-set we use consists of $N = 4,096$ examples extracted from raw interferograms, each of 128×128 resolution coming from the Sentinel-1 satellites, as described in (Rahman et al., 2022a). The satellite image covers a 250 by 160 km area around the Long Valley Caldera, an active volcano in Mammoth Lakes, California, from November 2014

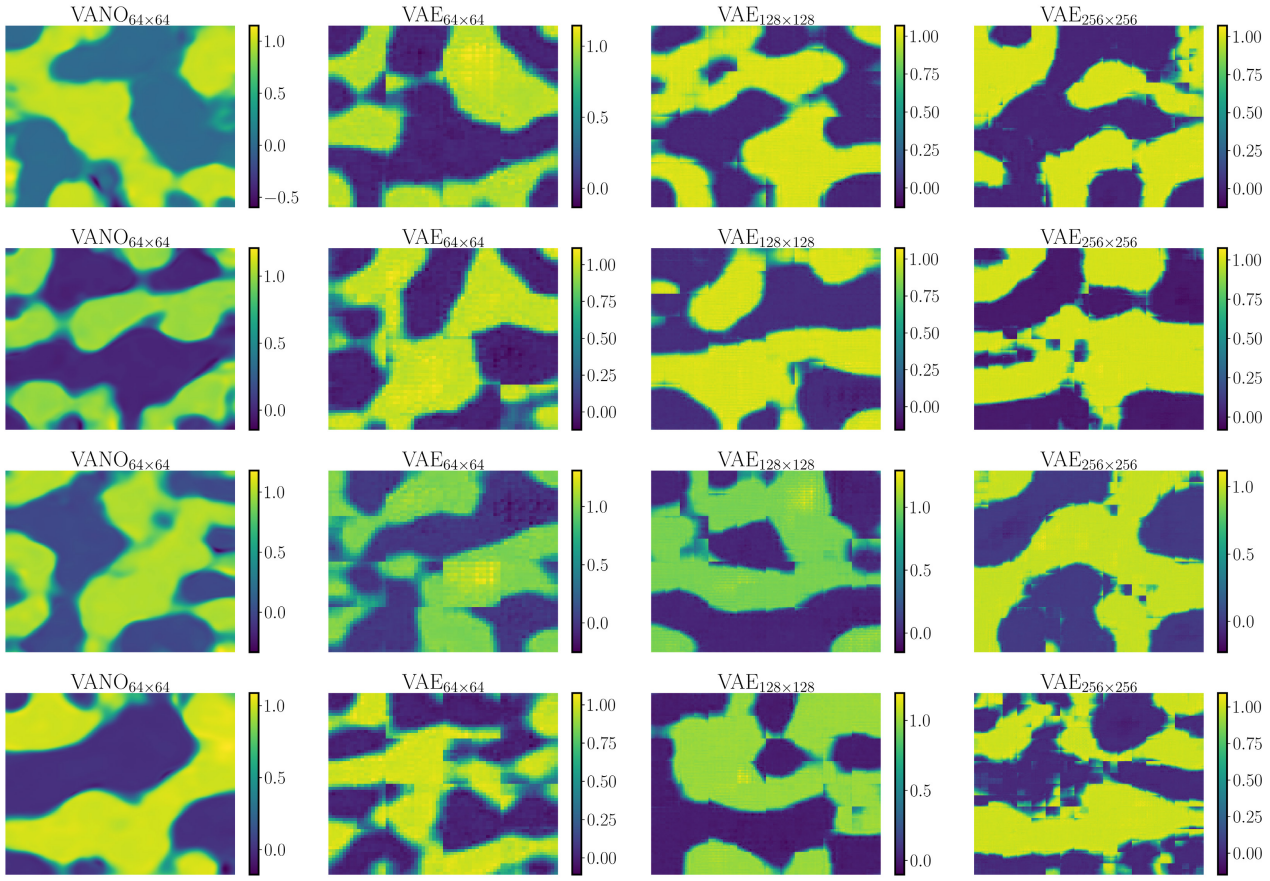

Figure 13. Cahn-Hilliard benchmark: On the far left column we provide 256x256 image samples from the VANO model trained on 64x64 resolution. On the other columns, we provide discretize-first VAE samples where the training resolution is indicated by their subscript, i.e. $\text{VAE}_{64\times64}$ indicates a samples coming from a VAE model trained on 64x64 resolution.

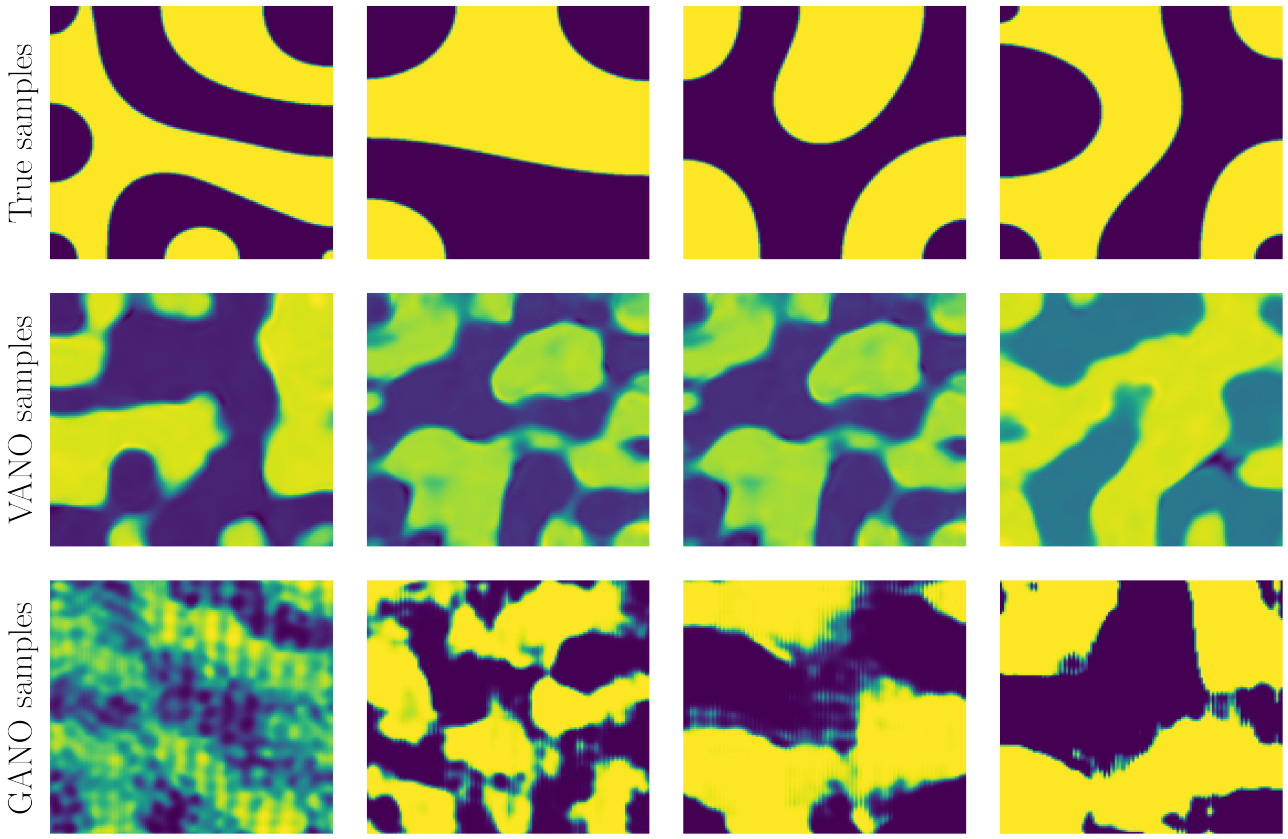

Figure 14. Cahn-Hilliard benchmark: Comparison of generated 128×128 samples from the ground truth dataset, VANO, and GANO.

to March 2022, using the InSAR Scientific Computing Environment (Rosen et al., 2012). The data is pre-processed as described in (Rahman et al., 2022a) to produce the training data-set. We train VANO on the entire data, as in (Rahman et al., 2022a).

Encoder: We employ a simple VGG-style convolutional encoder with 6 layers using 2×2 convolution kernels, stride of size two, (8, 16, 32, 64, 128, 256) channels per layer, and Gelu activation functions.

Decoder: We employ a nonlinear decoder parameterized by an 8-layer deep MLP network with 512 neurons per layer and split concatenation conditioning. We also employ the Multi-resolution Hash Encoding put forth by (Müller et al., 2022) in order to capture the multi-resolution structure in the target functional signals (see (Müller et al., 2022) for the default hyper-parameter settings).

Training Details: We consider a latent space dimension of $n = 256$, $S = 4$ Monte Carlo samples for evaluating the expectation of the reconstruction part of the loss and a KL loss weighting factor $\beta = 10^{-4}$. We train the model using the Adam optimizer (Kingma & Ba, 2014) with random weight factorization (Wang et al., 2022b) for $20,000$ training iterations with a batch size of 16 and a starting learning rate of 10^{-3} with exponential decay of 0.9 every 1,000 steps.

GANO Training Setup: We use the implementation from the official repository of the GANO paper [1] to train the model with the recommended hyper-parameter settings.

Evaluation: We evaluate the performance of our model using two metrics: the circular variance and the circular skewness, as explained in the Appendix Section E.3. Generated function samples are presented in Figure 17. We present reconstructions from the data-set in Figure 15 and new generated function samples in Figure 16.

E. Comparison Metrics

In this section we provide a description of different metrics used for evaluating the quality of our results.

E.1. Hilbert-Schmidt Norm

The Hilbert-Schmidt norm of an operator $T : \mathcal{H} \to \mathcal{H}$ on a Hilbert space $\mathcal{H}$ with orthonormal basis e_i is given by

$$\|T\|_{HS}^2 = \sum_i \langle Te_i, e_i \rangle. \tag{30}$$

If the operator T is self-adjoint with eigenvalues λ_i, this can also be written as

$$\|T\|_{HS}^2 = \sum_i \lambda_i^2. \tag{31}$$

Note that when $\mathcal{H}$ is a finite dimensional Hilbert space, this is equivalent to the standard 2 (Frobenius) norm for operators (matrices).

Since covariance operators for Gaussian measures always have finite Hilbert-Schmidt norm, we measure the distance of the two mean-zero Gaussian measures in the Gaussian random field example via the Hilbert-Schmidt norm of their difference. We approximate this via the approximations of the covariance operators in the discretization space $\mathbb{R}^{128\times128}$,

$$C = \sum_{i=1}^{n_{eig}} \lambda_i \tilde{\varphi}_i \tilde{\varphi}_i^\top, \quad \hat{C} = \sum_{i=1}^{n} \tilde{\tau}_i \tilde{\tau}_i^\top, \tag{32}$$

where $\tilde{\phi}_i, \tilde{\tau}_i \in \mathbb{R}^{128}$ are the evaluations of the functions ϕ_i and τ_i along the measurement points used in the experiment. The normalized Hilbert-Schmidt norm of the difference of the true covariance operators is then approximated as the Frobenius norm of the difference of their approximations divided by the Frobeinus norm of the true covariance C.

[1] https://github.com/kazizzad/GANO

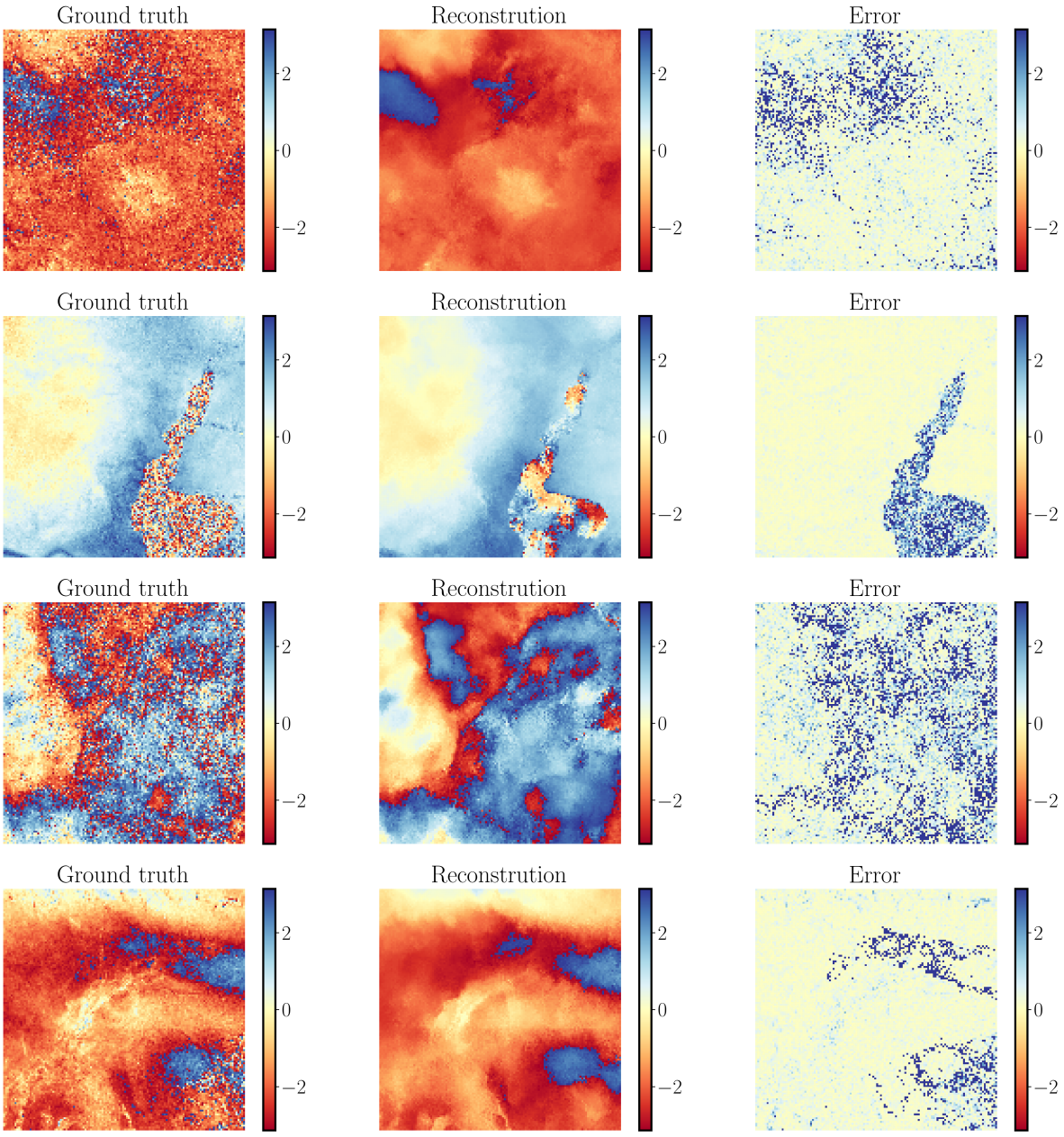

Figure 15. InSar benchmark: Left: Ground truth functions samples from the data-set, Middle: Linear VANO reconstruction, Right: Absolute error.

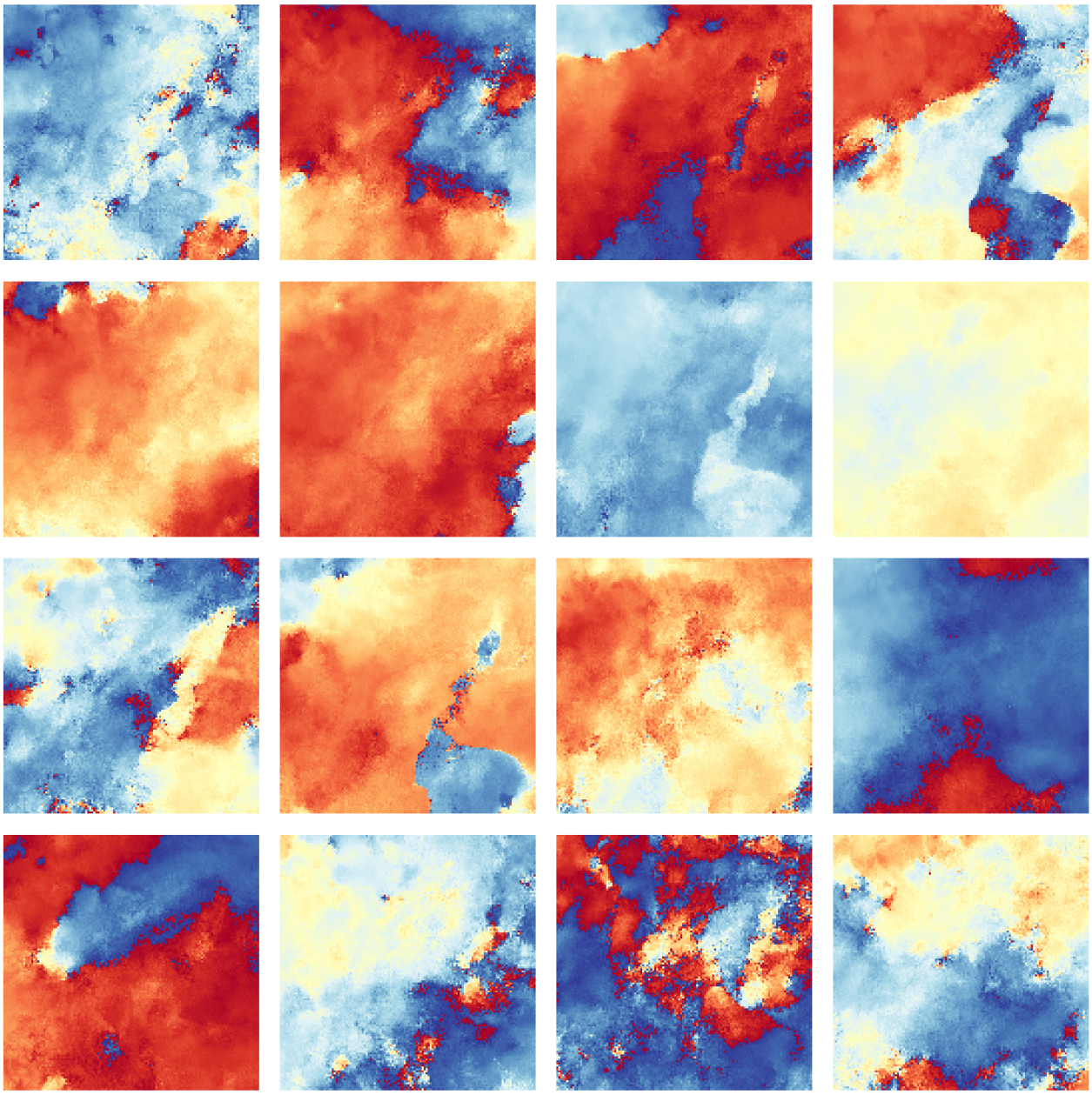

Figure 16. InSar benchmark: VANO generated function samples for InSAR Interferograms.

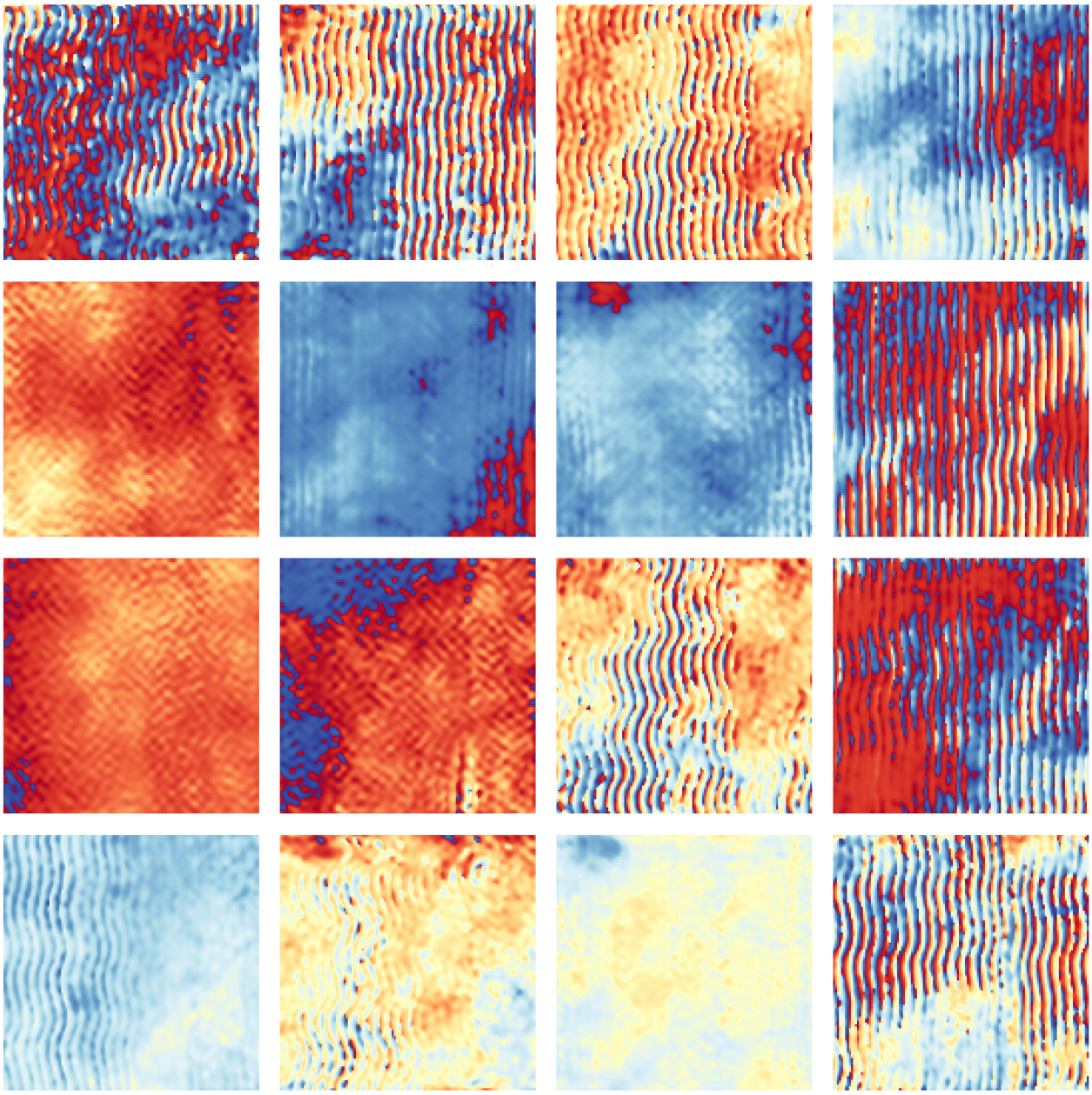

Figure 17. InSar benchmark: GANO (Rahman et al., 2022a) generated function samples for InSAR Interferograms.

E.2. Generalized Maximum Mean Discrepancy

For measuring the distance between ground truth and learned distributions, we choose to use a version of the Maximum Mean Discrepancy (MMD) distance. Given a probability distribution on a set $\mathcal{X}$ and a characteristic kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ (Sriperumbudur et al., 2011), the *kernel mean embedding* (Muandet et al., 2017) is a map from probability measures μ on $\mathcal{X}$ into the Reproducing Kernel Hilbert Space (RKHS) associated with k, $\mathcal{H}_k$ given by

$$\hat{\mu}_k := \int_{\mathcal{X}} k(\cdot, x) \, \mathrm{d}\mu(x). \tag{33}$$

Note that if μ is an empirical distribution, that is, a sum of delta measures

$$\mu = \frac{1}{N} \sum_{i=1}^{N} \delta_{x_i},$$

then the kernel mean embedding is given by

$$\hat{\mu}_k = \frac{1}{N} \sum_{i=1}^{N} k(\cdot, x_i). \tag{34}$$

Given two probability measures, μ and ν on a set $\mathcal{X}$, we can define the MMD distance between them as the distance between their kernel mean embeddings in the RKHS $\mathcal{H}_k$,

$$\mathrm{MMD}_k(\mu, \nu) = \|\hat{\mu}_k - \hat{\nu}_k\|^2_{\mathcal{H}_k}. \tag{35}$$

When both μ and ν are empirical distributions on points $\{x_i\}_{i=1}^N$ and $\{y_j\}_{j=1}^M$, respectively, the MMD can be evaluated as

$$\|\hat{\mu}_k - \hat{\nu}_k\|^2_{\mathcal{H}_k} = \frac{1}{N^2} \sum_{i,k=1}^{N} k(x_i, x_k) + \frac{1}{M^2} \sum_{j,\ell=1}^{M} k(y_j, y_\ell) - \frac{2}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} k(x_i, y_j). \tag{36}$$

While this is convenient for giving a notion of distance between empirical distributions corresponding to samples from a data-set and a generative model, it can be sensitive to the form of the kernel. For example, if a norm on $\mathcal{X}$ is used in a Gaussian kernel with a length-scale σ,

$$k_\sigma(x, y) = \exp\left(\frac{1}{2\sigma^2} \|x - y\|^2_{\mathcal{X}}\right), \tag{37}$$

for large enough σ the kernel will see all data points as being roughly the same and the MMD for any two fixed empirical distributions will become arbitrarily small.

To mitigate this problem, the generalized MMD distance was proposed (Fukumizu et al., 2009), which instead of using a single kernel uses a family of kernels $\mathcal{F}$ and defines a (pseudo-)metric between probability measures as

$$\mathrm{GMMD}_{\mathcal{F}}(\mu, \nu) := \sup_{k \in \mathcal{F}} \mathrm{MMD}_k(\mu, \nu). \tag{38}$$

As long as one of the kernels in $\mathcal{F}$ is characteristic, this defines a valid distance (Fukumizu et al., 2009).

In our experiments, we use the GMMD as a measure of distance of distributions with the family of kernels

$$\mathcal{F} = \left\{ k_\sigma \mid k_\sigma(x, y) = \exp\left(\frac{1}{2\sigma^2} \|x - y\|^2_{\mathcal{X}}\right), \ \sigma_- \leq \sigma \leq \sigma_+ \right\}. \tag{39}$$

Empirically, we find that the σ giving the largest MMD lies within the interval $\sigma_- = .1$ and $\sigma_+ = 20$ for all experiments, and therefore use a mesh of σ in this interval to approximate this $GMMD$. In Figure 18 we plot an example of the MMD for varying σ between 512 function samples from the 2D Gaussian densities data-set and those generated from the VANO model.

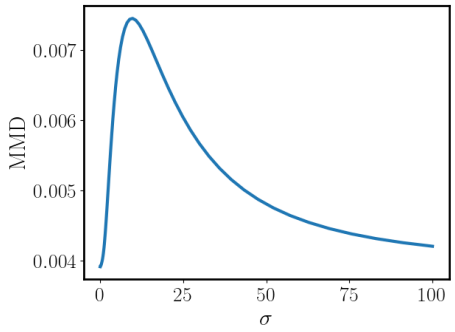

Figure 18. An example of MMDs computed over a range of lengthscales σ between the ground truth 2D Gaussian density data-set and the data-set generated by VANO.

Table 4. Computational cost for training all the models considered in this manuscript: We present the wall clock time *in seconds* that is needed to train each model on a single NVIDIA RTX A6000 GPU.

BENCHMARK	VAE	VANO (LINEAR)	VANO (NONLINEAR)	GANO
GRF	-	53	-	-
2D GAUSSIAN DENSITIES	-	67	198	-
CAHN-HILLIARD (64×64)	43	-	1,020	-
CAHN-HILLIARD (128×128)	55	-	-	-
CAHN-HILLIARD (256×256)	166	-	-	-
INSAR INTERFEROGRAM	-	-	11,820	42,060

E.3. Circular Variance and Skewness

The circular variance and skewness are moments of circular random variables, see (Rahman et al., 2022a), used to evaluate the quality of generated angular valued functions. For N random variables given by angles, $\{\theta_j\}_{j=1}^N$, let $z_p = \sum_i^N e^{ip\theta_j}$ with $i = \sqrt{-1}$. Define $\varphi_p = \arg(z_p)$ where arg is the complex argument function (returns the angle of a complex number to the real axis) and let $R_p = |z_p|/N$. The circular variance is then defined by $\sigma = 1 - R_1$ and the skewness by $s = \frac{R_2 \sin(\varphi_2 - 2\varphi_1)}{(1-R_1)^{3/2}}$.

F. Trainable Parameters and Computational Cost

We present the training time in seconds for each experiment and model in the manuscript in Table 4 as well as the total number of trainable parameters in Table 5.

Table 5. Total number of trainable parameters for all the models considered in this manuscript.

BENCHMARK	VAE	VANO (LINEAR)	VANO (NONLINEAR)	GANO
GRF	-	107,712	-	-
2D GAUSSIAN DENSITIES	-	85,368	89,305	-
CAHN-HILLIARD (64×64)	187,000	-	341,000	-
CAHN-HILLIARD (128×128)	485,000	-	-	-
CAHN-HILLIARD (256×256)	1,667,000	-	-	-
INSAR INTERFEROGRAM	-	-	11,130,420	48,827,763