Generative Modeling of Random Fields from Limited Data via Constrained Latent Flow Matching

James E. Warner[*]
NASA Langley Research Center
Hampton, VA 23681
james.e.warner@nasa.gov

Tristan A. Shah[*]
Texas Tech. University
Lubbock, TX 79409
trisshah@ttu.edu

Patrick E. Leser
NASA Langley Research Center
Hampton, VA 23681
patrick.e.leser@nasa.gov

Geoffrey F. Bomarito
NASA Langley Research Center
Hampton, VA 23681
geoffrey.f.bomarito@nasa.gov

Joshua D. Pribe
Analytical Mechanics Associates
Hampton, VA 23681
joshua.pribe@nasa.gov

Michael C. Stanley
Analytical Mechanics Associates
Hampton, VA 23681
michael.c.stanley@nasa.gov

Abstract

Deep generative models are promising tools for science and engineering, but their reliance on abundant, high-quality data limits applicability. We present a novel framework for generative modeling of random fields (probability distributions over continuous functions) that incorporates domain knowledge to supplement limited, sparse, and indirect data. The foundation of the approach is latent flow matching, where generative modeling occurs on compressed function representations in the latent space of a pre-trained variational autoencoder (VAE). Innovations include the adoption of a function decoder within the VAE and integration of physical/statistical constraints into the VAE training process. In this way, a latent function representation is learned that yields continuous random field samples satisfying domain-specific constraints when decoded, even in data-limited regimes. Efficacy is demonstrated on two challenging applications: wind velocity field reconstruction from sparse sensors and material property inference from a limited number of indirect measurements. Results show that the proposed framework achieves significant improvements in reconstruction accuracy compared to unconstrained methods and enables effective inference with relatively small training datasets that is intractable without constraints.

1 Introduction

Deep generative models (DGMs) have gained mainstream popularity by powering large language models (LLMs) [OpenAI et al., 2024] and text-to-image generators [Rombach et al., 2022], owing success to hardware and software breakthroughs, and, importantly, proliferation of massive training datasets. For science and engineering applications, data can be difficult and expensive to obtain, and the quantities being modeled are often infinite-dimensional functions of space and/or time, further exacerbating the data challenges. In order to fully leverage the power of DGMs in these scenarios,

[*] These authors contributed equally to this work.

Preprint. Under review.

more research is needed on approaches to supplement the training of these models using domain knowledge to offset the data limitations.

This paper considers the challenge of learning to sample from distributions of continuous functions in the absence of large, dense training datasets. These *random fields* are ubiquitous in science and engineering for modeling randomness in physical processes that vary spatially and/or temporally. For example, random fields can model the distribution of material properties in heterogeneous media [Ostoja-Starzewski, 1998], the velocity field in a turbulent flow [Mikulevicius and Rozovskii, 2004], or the temperature profiles for climate patterns [Guillot et al., 2015]. Training data for DGMs in these settings can be limited in three (not necessarily mutually exclusive) ways: 1) only relatively few total training data points are available, 2) each measurement may only provide partial/sparse information, and 3) only indirect measurements (i.e., of a related random field) are possible. Depending on data availability, two common scientific problems are the *reconstruction* of random fields from sparse measurements and *inference*[2] of unobserved random fields from indirect (and potentially limited number of) data.

As DGM research has progressed from single-step (e.g., generative adversarial networks (GANs) [Goodfellow et al., 2014], variational autoencoders (VAEs) [Kingma and Welling, 2013]) to multi-step (e.g., score-based/diffusion models [Ho et al., 2020, Nichol and Dhariwal, 2021], flow matching [Lipman et al., 2022, Tong et al., 2023]) to more recent latent-space models (e.g., latent diffusion [Rombach et al., 2022], latent flow matching (LFM) [Dao et al., 2023]), there has been a parallel research effort to tailor these approaches for science and engineering. A common focus is imposing constraints on DGMs to supplement training data with domain knowledge. These constraints are most commonly differential equation residuals, popularized by physics-informed neural networks (PINNs) [Raissi et al., 2019], but statistical constraints have also been considered [Wu et al., 2020]. Single-step DGMs offer a relative ease of incorporating constraints and a natural setting to output continuous functions but suffer from training instability and mode-collapse Saxena and Cao [2021]. Multi-step DGMs offer improved sample quality and stability but tend to work in a discretized (pixelized) space, and care is needed to enforce constraints on intermediate, noisy samples during training [Bastek et al., 2025].

Here, we propose a constrained LFM (c-LFM) approach for modeling random fields with limited data that combines the ease of constraint incorporation of single-step methods with improved stability and sample quality of multi-step methods. LFM leverages a continuous normalizing flow in the latent space of a pre-trained VAE to learn a mapping from prior distribution to complex latent variable distribution. We modify the VAE in two important ways to facilitate sampling continuous functions from sparse and limited data. First, a function decoder based on deep operator networks (DeepONets) [Lu et al., 2021] is used to produce samples that can be queried and differentiated anywhere in space/time during training (for imposing constraints) or sampling (for interpolation and extrapolation). Second, a modified VAE loss with residuals of physical or statistical constraints is used to learn a latent space representing functions that adhere to available domain knowledge. To the authors' knowledge, this work is the first to incorporate constraints into LFM for limited training data regimes, offering the following practical advantages:

1. Models random fields, i.e., aleatory uncertainty as a distribution over functions (rather than the simpler, more common estimation of epistemic uncertainty about a single function).
2. Remains effective even when only sparse or indirect data is available *at train time*.
3. Produces samples that are continuous functions of space and/or time.

We demonstrate both reconstruction and inference of random fields with c-LFM on both illustrative toy problems and real-world applications.

2 Related Work

Relevant work on tailoring DGMs to perform probabilistic modeling for science and engineering is reviewed here, separated based on category of DGM. As our c-LFM approach 1) models random fields (distributions of continuous functions), 2) operates in the limited data regime at train time, and 3) produces continuous functions of space/time, we focus on these aspects.

[2] We use "inference" to refer to solving inverse problems involving random fields, and mainly refer to the "inference" stage of DGMs as "sampling" to avoid confusion.

Single-Step Models exhibit high-speed sample generation and relative ease of generating constrained functions of space/time. GANs were used to model fluid flow using covariance constraints [Wu et al., 2020] and novel discriminator architectures [Xie et al., 2018] to encourage physical consistency, albeit in a data-driven regime. Physics constraints imposed in adversarial training procedures were used to infer unknown deterministic functions from partial/noisy data [Yang and Perdikaris, 2019, Daw et al., 2021]. Methods for estimating random fields from limited data by incorporating physics knowledge were proposed using GANs [Yang et al., 2018] and VAEs [Zhong and Meidani, 2023, Rodrigo-Bonet and Deligiannis, 2024], while Rodrigo-Bonet and Deligiannis [2024] considered discrete, rather than continuous, quantities of interest.

Multi-Step Models improve upon single-step methods in stability and sample quality, but typically operate in discrete (pixel) space and require special care when enforcing constraints on intermediate samples during training. Bastek et al. [2025] proposed a physics-informed diffusion model that uses scaling and annealing to enforce constraints but requires finite difference to evaluate constraints due to the discrete nature of the diffusion model. Other diffusion model-based approaches apply physics constraints at sampling time instead [Christopher et al., 2024, Jacobsen et al., 2024, Shu et al., 2023, Huang et al., 2024] or use conditional sample generation with no constraints [Dasgupta et al., 2025], and thus do not apply when training data is sparse/indirect. Flow matching with physics-controls were investigated by Holzschuh and Thuerey [2024] for inferring deterministic functions.

Latent-Space Models have been tailored to solve science and engineering problems. Latent diffusion models were used to solve inverse problems in Song et al. [2024] for medical imaging and in Shmakov et al. [2023] for high energy physics. The former used optimization to promote data consistency during sampling while the latter introduced a novel “variational latent diffusion" model with physics-informed consistency loss. Du et al. [2024] used a latent diffusion model for turbulence reconstruction and super resolution, but used a conditional neural field approach in a data-driven regime rather than relying on physics information.

3 Background

A random field can be defined as the n-dimensional mapping, $\mathbf{U} : \mathcal{X} \times \Omega \rightarrow \mathbb{R}^n$, where $\mathcal{X} \subset \mathbb{R}^{d_x}$ is the physical domain representing spatial/temporal coordinates ($1 \leq d_x \leq 4$), and Ω is a sample space accounting for randomness in $\mathbf{U}$. For each fixed $\hat{\omega} \in \Omega$, $\mathbf{u}(\mathbf{x}) \equiv \mathbf{U}(\mathbf{x}, \hat{\omega})$ defines a deterministic function[3] over $\mathcal{X}$, while each fixed $\hat{\mathbf{x}} \in \mathcal{X}$, $\mathbf{U}(\hat{\mathbf{x}}, \omega)$ represents a random variable in $\mathbb{R}^n$. The ability to generate samples of random fields, $\mathbf{u}(\mathbf{x})$, that are consistent with available data, physically-admissible, and can be evaluated at arbitrary $\mathbf{x}$ is of great importance to science and engineering applications.

A common challenge of applying DGMs in this context is sparse and limited training data. More concretely, let $\mathcal{T}(\mathbf{u}) = \mathbf{y} \in \mathbb{R}^{m \times n}$ be a measurement operator where $\mathbf{y} = \{\mathbf{u}(\mathbf{x}), \mathbf{x} \in \mathbf{X}^s\}$ are finite dimensional observations of $\mathbf{u}$ and $\mathbf{X}^s$ is a set of m spatiotemporal coordinates where measurements are available. Assume a collection of N independent observations are available for training, represented by the empirical probability distribution, $p(\mathbf{y}) \equiv \{\mathbf{y}^{(i)}\}_{i=1}^{N}$. There are many practical situations where m is small (e.g., sparse measurements in space and/or time when few sensors are available) or N is small (e.g., measurements are expensive and/or time consuming). Furthermore, many applications only allow *indirect* observations, i.e., measurements of $\mathbf{U}$ are leveraged to perform inference on a related random field of interest, $\mathbf{V}(\mathbf{x}, \omega)$. We focus on the challenges above but do not address *noisy* measurements, an important limitation left to future work.

While science and engineering contend with smaller datasets than language and image applications, domain knowledge in the form of statistical and physical constraints can often be exploited, i.e.,

$$R\left(\mathbf{U}(\mathbf{x}, \omega)\right) = 0, \tag{1}$$

$$F\left(\mathbf{u}(\mathbf{x}), \mathbf{v}(\mathbf{x})\right) = 0, \tag{2}$$

where $\mathbf{v}(\mathbf{x}) \equiv \mathbf{V}(\mathbf{x}, \hat{\omega})$. R enforces a known probabilistic metric and operates on random variables, e.g., prescribing a known mean, μ_u, as $R\left(\mathbf{U}\right) = \mathbb{E}[\mathbf{U}] - \mu_u$. F typically represents the residual of known partial/ordinary differential equations that operate on deterministic functions, e.g., $F(\mathbf{u}, \mathbf{v}) =$

[3] We largely follow the convention of denoting random quantities with uppercase letters and deterministic realizations of random quantities with lowercase letters. Vector-valued quantities are identified by bold fonts.

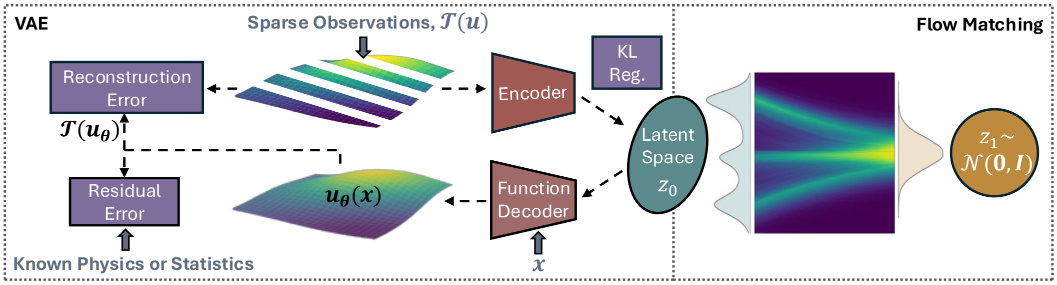


Figure 1: **c-LFM for modeling random fields from limited data**. A VAE with function decoder learns a latent representation of the continuous random field, $\mathbf{U}(\mathbf{x}, \omega)$, from sparse observations. VAE loss terms are shown in purple boxes, where a residual error supplements limited data with statistical/physical constraints. Flow matching enables latent variable sampling.

$\mathcal{N}_x(\mathbf{u}, \mathbf{v})$, where $\mathcal{N}_x$ is a differential operator. Two data-limited science and engineering problems demonstrating the use of these constraints are described next.

3.1 Problem 1: Random Field Reconstruction from Sparse Data

- **Goal:** Learn the probability distribution, $p(\mathbf{u})$, over functions $\mathbf{u}(\mathbf{x})$ given the observations, $\{\mathbf{y}^{(i)}\}_{i=1}^{N}$, and statistical constraint(s), $R(\mathbf{U})$. A DGM is sought that generates samples of functions $\mathbf{u}$ that satisfy R and are consistent with observations.
- **Data Characteristics:** There are sufficient numbers of observations ($N >> 1$) but each observation only provides sparse information on $\mathbf{u}$, i.e., m is small.
- **Target Application:** Wind velocity field estimation where probabilistic assessments of wind flows are required continuously over a spatial region and short time horizon, but wind measurements are only available on a relatively coarse spatiaotemporal grid.

3.2 Problem 2: Random Field Inference from Indirect Data

- **Goal:** Learn the joint probability distribution, $p(\mathbf{u}, \mathbf{v})$, over functions $\mathbf{u}(\mathbf{x})$ and $\mathbf{v}(\mathbf{x})$ given observations, $\{\mathbf{y}^{(i)}\}_{i=1}^{N}$, and physical constraints relating the functions, $F(\mathbf{u}, \mathbf{v})$. A DGM is sought that generates pairs of functions $(\mathbf{u}(\mathbf{x}), \mathbf{v}(\mathbf{x}))$ that satisfy F and where $\mathbf{u}$ is consistent with observations.
- **Data Characteristics:** The primary quantity of interest, $\mathbf{v}(\mathbf{x})$, is unobservable. A related quantity, $\mathbf{u}(x)$, is observed with sufficient numbers of sensors per observation ($m >> 1$) but the total number of observations may be limited, i.e., N may be small.
- **Target Application:** Material property characterization where the property varies spatially within a test article and also between different test articles, e.g., due to manufacturing variability. Experimental techniques provide dense measurements per test but relatively few tests can be performed.

3.3 Random Field Vs. Deterministic Function Estimation

There is an important distinction between the reconstruction and inference of random fields (the focus of this work) versus deterministic functions (the focus of a majority of existing work [e.g., Yang and Perdikaris, 2019, Yang et al., 2019, Shu et al., 2023]). For random fields, we leverage DGMs to model the *aleatory* uncertainty (intrinsic randomness) as a probability distribution over deterministic functions. When estimating a deterministic function, the DGM quantifies *epistemic* uncertainty (lack of knowledge about the true function) due to limited and/or noisy data. Mathematically, most previous work has approximated conditional probability distributions for both reconstruction, $p(\mathbf{u}|\mathcal{T}(\mathbf{u}) = \mathbf{y}^{(i)})$ [Du et al., 2024, Tran et al., 2020], and inference, $p(\mathbf{v}|\mathcal{T}(\mathbf{u}) = \mathbf{y}^{(i)})$ [Dasgupta et al., 2025, Shmakov et al., 2023], while we seek the marginal/joint distributions $p(\mathbf{u})$ and $p(\mathbf{u}, \mathbf{v})$.

4 Methodology

An overview of the framework is shown in Figure 1. The foundation of the c-LFM approach involves generative modeling in the latent space of a pretrained VAE using flow matching. The VAE encodes discrete (potentially sparse) data into latent variables that are then transformed into continuous function samples using a function decoder. Known physical and statistical constraints are incorporated into the VAE loss function through a residual error. The latent space provides a compressed (typically low-dimensional) function parameterization that leads to straightforward training and sampling with flow matching. See Appendix A.2.1 for training and sampling algorithms.

4.1 VAE with Function Decoder and Constraints

Following Kingma and Welling [2013], we assume samples, $\mathbf{u}(\mathbf{x})$, are generated from the conditional distribution, $p(\mathbf{u}|\mathbf{z})$, where $\mathbf{z} \in \mathbb{R}^{d_z}$ are d_z unobserved (latent) variables described by a prior distribution, $p(\mathbf{z})$. The posterior distribution of the latent variables is $q(\mathbf{z}|\mathbf{y})$, where $\mathbf{y} = \mathcal{T}(\mathbf{u})$ are finite observations of function $\mathbf{u}$, as described in Section 3. Note that $\mathbf{z}$ can be viewed as accounting for stochasticity in the random field, i.e., $\mathbf{U}(\mathbf{x}, \omega) = \mathbf{U}(\mathbf{x}, \mathbf{Z}(\omega))$ and we seek to estimate $p(\mathbf{u}) = \int p(\mathbf{u}|\mathbf{z})p(\mathbf{z})d\mathbf{z}$.

VAEs approximate the conditional probability distributions with a neural network-based encoder and decoder. The encoder network, $\mathcal{E}_{\boldsymbol{\phi}}$, is parametrized by $\boldsymbol{\phi}$ and maps discrete observations to latent space, $\mathbf{z} = \mathcal{E}_{\boldsymbol{\phi}}(\mathbf{y}) \sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{y})$. The reparameterization trick [Kingma and Welling, 2013] is used such that latent variables are sampled as $\mathbf{z} = \boldsymbol{\mu}_{\boldsymbol{\phi}} + \epsilon \circ \boldsymbol{\sigma}_{\phi}$, where $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $\circ$ denotes an element-wise product, and $\boldsymbol{\mu}_{\boldsymbol{\phi}}, \boldsymbol{\sigma}_{\boldsymbol{\phi}}$ are the mean and standard deviation of $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{y})$ predicted by the decoder.

To decode a latent sample, $\mathbf{z}^{(j)}$, into a continuous function, $\mathbf{u}^{(j)}(\mathbf{x}) \sim p_{\boldsymbol{\theta}}(\mathbf{u}|\mathbf{z} = \mathbf{z}^{(j)})$, our first modification is to implement a function decoder, $\mathcal{D}_{\boldsymbol{\theta}}$, using DeepONets [Lu et al., 2021]:

$$\mathcal{D}^u_{\boldsymbol{\theta}}(\mathbf{z}, \mathbf{x}) \equiv \sum_{k=1}^{p} b^u_k(\mathbf{z}; \boldsymbol{\psi}) t^u_k(\mathbf{x}; \boldsymbol{\xi}). \tag{3}$$

Here, $\boldsymbol{\theta} = [\boldsymbol{\psi}, \boldsymbol{\xi}]$ and the decoder output is the dot product of *branch* network, $\mathbf{b}^u_{\boldsymbol{\psi}}$, which receives latent variables, and *trunk* network, $\mathbf{t}^u_{\boldsymbol{\xi}}$, which receives spatial/temporal coordinates. This architecture decomposes $\mathbf{U}(\mathbf{x}, \omega)$ into stochastic and deterministic components, as is common in random field modeling [Karhunen, 1947, Loeve, 1955, Xiu, 2010], and provides a simple method of evaluating samples arbitrarily in $\mathbb{R}^{d_x}$. Note that in principle our framework should be compatible with other function parameterizations as well. For the case of random field inference (Section 3.2), a second decoder for $\mathbf{v}(\mathbf{x})$, $\mathcal{D}^v_{\boldsymbol{\theta}}(\mathbf{z}, \mathbf{x})$, is defined similarly.

Since we assume observations, $\mathbf{y}$, provide only partial and/or indirect information for generative modeling, our second VAE modification updates the evidence lower bound (ELBO) loss, $\mathcal{L}_{ELBO}$, to incorporate statistical and physical constraints. For a single batch of data, $\{\mathbf{y}^{(j)}\}$, the updated loss is

$$\begin{aligned}\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}; \mathbf{y}^{(j)}) &= \mathcal{L}_{ELBO}(\boldsymbol{\phi}, \boldsymbol{\theta}; \mathbf{y}^{(j)}) + \lambda_r \|R(\mathcal{D}^u_{\boldsymbol{\theta}}(\mathbf{z}^{(j)}, \mathbf{x}))\|^2 \quad + \lambda_f \|F(\mathcal{D}^u_{\boldsymbol{\theta}}(\mathbf{z}^{(j)}, \mathbf{x}), \mathcal{D}^v_{\boldsymbol{\theta}}(\mathbf{z}^{(j)}, \mathbf{x}))\|^2 \\ &= \underbrace{\|\mathcal{T}(\mathcal{D}^u_{\boldsymbol{\theta}}(\mathbf{z}^{(j)}, \mathbf{x})) - \mathbf{y}^{(j)}\|^2}_{\text{Reconstruction error}} \quad + \underbrace{\lambda_{kl} D_{KL}\left(q_{\boldsymbol{\phi}}(\mathbf{z}^{(j)}|\mathbf{y}^{(j)}), p(\mathbf{z})\right)}_{\text{KL regularization}} \\ &\quad + \underbrace{\lambda_r \|R(\mathcal{D}^u_{\boldsymbol{\theta}}(\mathbf{z}^{(j)}, \mathbf{x}))\|^2}_{\text{Statistics residual}} \quad + \underbrace{\lambda_f \|F(\mathcal{D}^u_{\boldsymbol{\theta}}(\mathbf{z}^{(j)}, \mathbf{x}), \mathcal{D}^v_{\boldsymbol{\theta}}(\mathbf{z}^{(j)}, \mathbf{x}))\|^2}_{\text{Physics residual}},\end{aligned} \tag{4}$$

where $\mathbf{z}^{(j)} \sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{y} = \mathbf{y}^{(j)})$, D_{KL} is the Kullback-Leibler (KL) divergence, and λ_r, λ_f, λ_{kl} are weights. The expected loss, $\mathbb{E}[\mathcal{L}] = \frac{1}{B}\sum_{j=1}^{B} \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}; \mathbf{y}^{(j)})$, is minimized during training with B batches.

Equations 1 and 2 are intended to hold $\forall \mathbf{x}$. This is commonly implemented by estimating residual terms over collocation points $\{\tilde{\mathbf{x}}^{(c)}\}_{c=1}^{C}$ [Raissi et al., 2019]. We choose these points randomly at each training iteration, $\tilde{\mathbf{x}}^{(c)} \sim \text{Uniform}(\mathcal{X})$. While simultaneous statistical and physical constraints is possible, we apply one at a time (statistical for reconstruction and physical for inference). Details regarding constraint residual calculations are provided in Appendix A.2.2.

4.2 Flow Matching

Latent generative modeling approaches like LFM typically deemphasize D_{KL} to encourage regularization while enabling a more expressive latent space, e.g., $\lambda_{kl} \approx 10^{-6}$ [Rombach et al., 2022]. Then, multi-step generative models can be trained to sample the posterior $p(\mathbf{z}|\mathbf{y})$ instead of simply decoding samples from the prior $p(\mathbf{z})$, since those distributions will generally differ. The LFM approach used here and described below follows the originally proposed method [Dao et al., 2023].

Let $\mathbf{z}_0 = \mathcal{E}_{\boldsymbol{\phi}}(\mathbf{y})$ be an encoded observation and $\mathbf{z}_1$ denote random noise, i.e., $\mathbf{z}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Flow matching posits the following ordinary differential equation to transform samples from reference (noise) distribution to source (latent) distribution:

$$\frac{d\mathbf{z}_t}{dt} = \boldsymbol{\nu}(\mathbf{z}_t, t) \tag{5}$$

where time $t \in [0,1]$ and $\boldsymbol{\nu} : \mathbb{R}^{d_z} \times [0,1] \to \mathbb{R}^{d_z}$ is the velocity to drive the flow between distributions. We choose the simple but effective linear interpolation between reference and source samples, $\mathbf{z}_t \equiv (1-t)\mathbf{z}_0 + t\mathbf{z}_1$, yielding a constant velocity ODE with $\boldsymbol{\nu} = \mathbf{z}_1 - \mathbf{z}_0$.

A neural network approximation for velocity, $\boldsymbol{\nu}_\eta$ with parameters η, is introduced and trained using

$$\eta^* = \arg\min_{\eta} \mathbb{E}_{t,\mathbf{z}_t} \|\mathbf{z}_1 - \mathbf{z}_0 - \boldsymbol{\nu}_\eta(\mathbf{z}_t, t)\|_2^2, \tag{6}$$

where $t \sim \text{Uniform}(0,1)$ and $\mathbf{z}_t$ is sampled by interpolating between $\mathbf{z}_0 = \mathcal{E}_{\boldsymbol{\phi}}(\mathbf{y})$ for $\mathbf{y} \sim p(\mathbf{y})$ and $\mathbf{z}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ given t. After training, realizations of $\mathbf{z}_0$ can be generated by numerically integrating Equation 5 with random noise as the initial condition and right-hand side ν_{η^*}. Resulting latent variables can be then be decoded to produce continuous samples of $\mathbf{u}(\mathbf{x})$ and/or $\mathbf{v}(\mathbf{x})$. See Algorithms 2 and 3 for more details on training and sampling, respectively.

5 Experiments

The proposed c-LFM approach is demonstrated for random field reconstruction (Section 3.1) and inference (Section 3.2), with each illustrated by simple toy demonstration and an application with real-world motivation. Code is provided at `https://github.com/nasa/random_field_c-lfm`.

5.1 Demonstration 1: Random Field Reconstruction from Sparse Data

We reconstruct a one-dimensional random field (i.e., stochastic process) using sparse observations and a statistical constraint. The unknown field is modeled by a Gaussian process (GP) with linear mean function and squared exponential covariance function:

$$\begin{aligned} u(x) &\sim \mathcal{GP}(\mu(x) = x, C(x, x')), \quad x \in [0,1] \\ C(x, x') &= \sigma^2 \text{exp}\left(-\frac{\|x - x'\|^2}{2l^2}\right), \end{aligned} \tag{7}$$

with variance, $\sigma^2 = 0.5$, and covariance length, $l = 0.1$. The goal is to learn to sample $u(x)$ for sparse ($m \leq 3$) equally-spaced sensors but with sufficient observations ($N = 1000$). A statistical constraint, R, imposes the true covariance structure during training to supplement the sparse data. The constraint residual was computed with $C = 50$ randomly chosen collocation points and weighted by $\lambda_r = 0.1$ during training.

Figure 2 shows the true GP samples and covariance and corresponding generated samples for $m = 1, 3$, with and without a covariance constraint. c-LFM begins to generate visually similar samples and covariance to the true field with just a single sensor, while standard LFM generates spatially constant samples, as expected. Both approaches accurately capture the mean for $m = 3$, while the no-constraint approach significantly underperforms the proposed approach in estimating covariance. While simple, this example illustrates the impact of statistical constraints on random field reconstruction from sparse data. See Appendix A.1.1 for more detail.

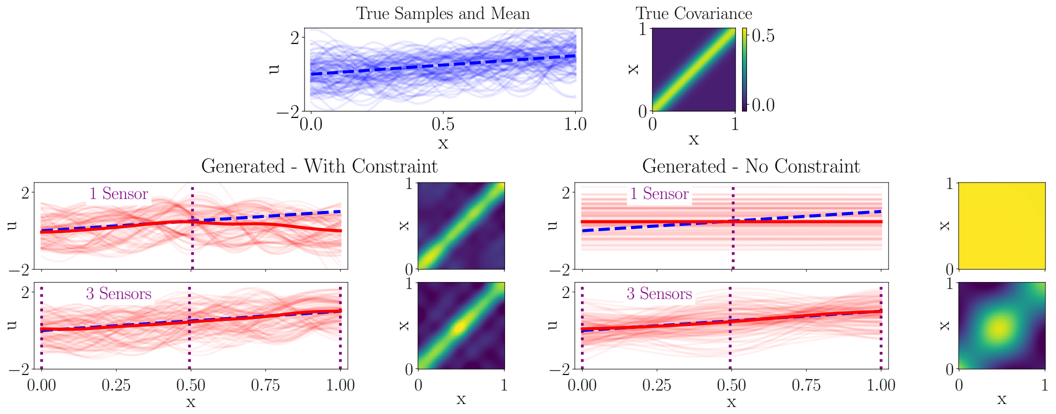


Figure 2: Demonstration of random field reconstruction from sparse data with c-LFM. (*Top*) True samples and statistics from a Gaussian Process along with (*Bottom*) generated samples and statistics using c-LFM with covariance constraint (left) versus standard LFM with no constraint (right). The statistical constraint allows for accurate covariance recovery from sparse sensors.

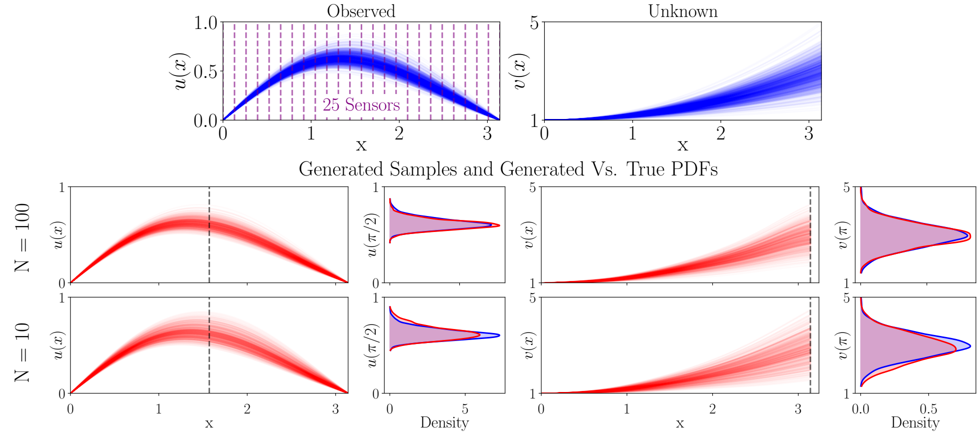


Figure 3: Demonstration of random field inference from indirect data with c-LFM. (*Top*) true samples of $u(x)$ (observed) and $v(x)$ (unknown) and (*Bottom*) generated samples along with true (blue) versus generated (red) pointwise distributions. A physical constraint incorporating the Poisson equation allows for inference of $v(x)$ from limitied observations of $u(x)$ only.

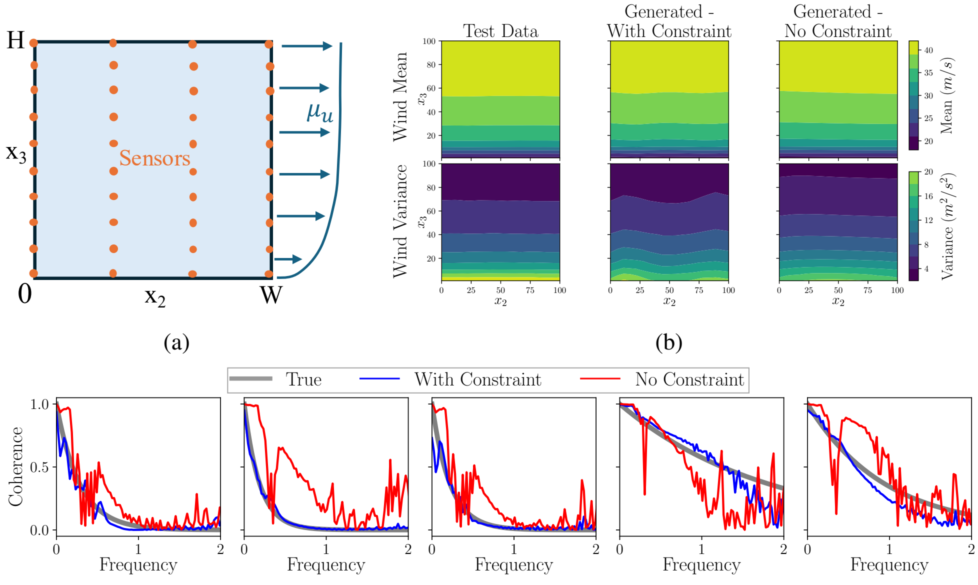
5.2 Demonstration 2: Random Field Inference from Limited and Indirect Data

A spatially-varying, random coefficient, $V(x, \omega)$, is estimated from observations of random field, $U(x, \omega)$, where samples of the two are related through the one-dimensional Poisson equation:

$$\begin{aligned} &\frac{d}{dx}\left(v(x)\frac{du}{dx}\right) = -f(x), \quad x \in [0, \pi], \\ &u(0) = 0, \ u(\pi) = 0 \end{aligned} \tag{8}$$

with $f(x) = \sin(x)$, $v(x) = 1 + \epsilon x^2$, and $\epsilon \sim \mathcal{N}(0.2, 0.05)$. The goal is to learn to sample from $p(u, v)$ with limited observations of $u(x)$ ($N \leq 100$) and sufficiently dense sensors ($m = 25$) using c-LFM. Synthetic data is generated by sampling ϵ, computing $v(x)$, and using numerical integration to solve for $u(x)$. The unknown random field, $V(x, \omega)$, is learned by incorporating Equation 8 into the loss via physical constraint F with weight, $\lambda_f = 0.001$, and $C = 50$ collocation points.

A summary of the results is shown in Figure 3, comparing samples of the true and generated random fields for $N = 100$ and $N = 10$. Sample-based empirical PDFs of $U(x = \pi/2)$ and $V(x = \pi)$ are shown to compare the point-wise distributions of the estimated versus true fields. Despite having no



(a)

(b)

(c)

Figure 4: Wind velocity estimation (a) diagram along with comparisons of (b) mean and variance wind fields and (c) coherence for the true wind test data versus generated samples (with and without coherence constraint).

direct observations, $V(x, \omega)$ is recovered effectively using c-LFM with physical constraint, allowing reasonably accurate inference with only $N = 10$ observations. We emphasize that inference of $V(x, \omega)$ is intractable with standard LFM since there is not direct training data. See Appendix A.1.2 for more detail.

5.3 Application 1: Wind Velocity Field Estimation

A real-world example of random field reconstruction is presented to demonstrate c-LFM in a higher dimensional setting with a more complex statistical constraint. The goal is to generate realistic samples of a wind velocity field, $\mathbf{U}(\mathbf{x}, \omega)$, from sparse spatio-temporal observations. Synthetic measurement data is generated via Monte Carlo simulation following a common wind engineering formulation Carassale and Solari [2006], decomposing a random wind field as

$$\mathbf{U}(\mathbf{x}, \omega) = \boldsymbol{\mu}_u(\mathbf{x}) + \mathbf{W}(\mathbf{x}, \omega), \tag{9}$$

where $\mathbf{x} = [x_1, x_2, x_3, t]$, $\boldsymbol{\mu}_u$ is a deterministic mean function describing a logarithmic profile versus altitude, and $\mathbf{W}(\mathbf{x}, \omega)$ is a zero-mean, stationary GP. The spatial and temporal correlation of $\mathbf{W}$ (and hence $\mathbf{U}$) is prescribed by the *coherence function*:

$$\text{Coh}(\mathbf{x}, \mathbf{x}', n) = \exp\left(-\frac{n\|\mathbf{c}^T(\mathbf{x} - \mathbf{x}')\|}{\|\mu_u(\mathbf{x}) - \mu_u(\mathbf{x}')\|}\right), \tag{10}$$

which generalizes spatial correlation to include frequency, n. Decay coefficients $\mathbf{c}$ control coherence decrease in each spatial direction. Samples of $\mathbf{U}$ were generated for prescribed mean and coherence using the spectral representation method [Shinozuka and Deodatis, 1991, Deodatis and Shields, 2025] implemented in `UQpy` [Olivier et al., 2020]. See Appendex A.1.3 for details.

This example focuses on reconstructing the U_1 velocity component over time in a two-dimensional plane, $[x_2, x_3, t] \times [0, 100]m \times [1, 100]m \times [0, 42.5]s$; see Figure 4 (a). The training data comprised $N = 5000$ velocity samples at 256 time steps and four equally-spaced vertical sensor arrays with ten sensors each, mimicking real-world wind profilers [Coulter and Muradyan, 2020, Shah et al., 2025]. c-LFM was applied with Equation 10 used as a statistical constraint for training, where the coherence residual was estimated based on the `pytorch` fast Fourier transform. Importantly, since the coherence is a normalized metric, c-LFM must balance learning the wind velocity mean and variance from the sparse data and spatial/temporal correlation from the statistical constraint.

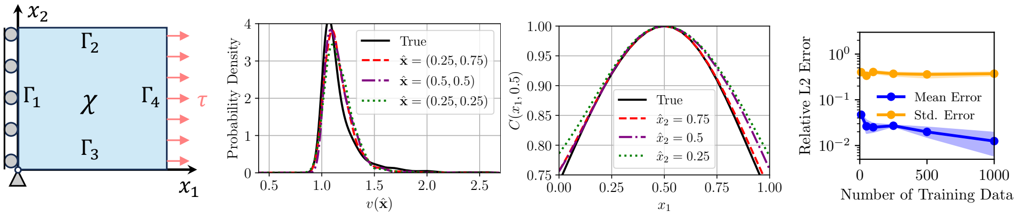
Figure 4 (b) shows mean and variance fields estimated from 1000 samples and compares generated samples with and without a coherence constraint to true samples from a test dataset. Generated samples for both cases are similar, accurately recovering the mean but slightly underestimate the variance near $x_3 = 0$ where test data variance increases rapidly. Figure 4 (c) compares true wind coherence to empirical estimates from generated samples at five spatial coordinate pairs. Standard LFM struggles while c-LFM accurately recovers the coherence from the statistical constraint. The coherence constraint also appears to help regularize the problem, improving the convergence of LFM training, see Appendix A.1.3 for details.

5.4 Application 2: Material Property Characterization

A real-world engineering inference problem is presented in which only indirect observations are available. The goal is to estimate non-observable, spatially-varying material property field, $V(\mathbf{x}, \omega)$, using measurements of the mechanical deformation, $\mathbf{U}(\mathbf{x}, \omega)$, induced by a deterministic tensile force. We consider the two-dimensional domain illustrated in Figure 5(a) where $\mathbf{x} \in [0,1] \times [0,1]$ and $\mathbf{u} \in \mathbb{R}^2$. Inference is facilitated through a second-order partial differential equation (PDE) that relates samples of $\mathbf{U}$ and V, $\mathcal{N}_x(\mathbf{u}, v) = 0$ (Equation 17). Additionally, there are homogenous Dirichlet boundary conditions, $\mathcal{B}_d(\mathbf{u})$, on Γ_1 and Neumann boundary conditions, $\mathcal{B}_n(\mathbf{u}, v)$, on Γ_4.

The ground truth material property was modeled as a lognormal random field with squared exponential correlation function. Synthetic training data for $\mathbf{U}$ were generated by randomly sampling V and numerically solving the governing PDE, as described in Appendix A.2. Measurements of $\mathbf{U}$ were made available on a (10×10) uniform grid spanning the domain with left column ($\mathbf{x} \in \Gamma_1$) removed, resulting in $m = 90$ sensors. To study the effect of limited data at train time, we vary the number of training data, $N = 10, 50, 100, 250, 500, 1000$. Three trials with different random seeds were conducted for each value of N. The physics constraint, $F(\mathbf{u}, v) = \mathcal{N}_x(\mathbf{u}, v) + \mathcal{B}_d(\mathbf{u}) + \mathcal{B}_n(\mathbf{u}, v)$, was evaluated with $C = 100$ and $\lambda_f = 10^{-6}$. See Appendix A.2 for details.

Figure 5(b) and (c) show the accuracy of inferred random material property field, $V(\hat{\mathbf{x}})$, for the case of $N = 1000$ training data in terms of the estimated probability density functions (PDFs) and spatial correlations throughout the domain. The PDFs estimated at three distinct locations match closely with the ground truth, which is shown as a solid black line, albeit with a thinner lower tail. There is also relatively close agreement between the estimated spatial correlation across three horizontal slices in the domain (Correlation$(x_1, 0.5)$) relative to the ground truth random field. The relative L2 error between true and estimated fields is shown in Figure 5(d) as a function of N, where it can be seen that the accuracy is comparable to the $N = 1000$ case until the errors increase noticeably for $N = 10$. Plots of individually generated samples are provided in Appendix A.1.4. The results demonstrate the ability of c-LFM to characterize unobservable random fields using physics constraints and limited, indirect data at train time, while inference would be intractable with an unconstrained approach.



(a) (b) (c) (d)

Figure 5: Material property characterization (a) diagram, the (b) PDF at three distinct spatial coordinates, $\hat{\mathbf{x}}$, and (c) correlation with respect to $V(0.5, \hat{x_2})$ along horizontal slices versus the true material property, and (d) the relative L2 error in mean and variance field vs. number of train data.

6 Conclusion

Inspired by science and engineering applications, a novel constrained LFM (c-LFM) approach for learning random fields from limited training data was introduced. By integrating domain knowledge constraints into a VAE with a DeepONet function decoder and leveraging stability and sample quality benefits of LFM, accurate random field reconstruction from sparse data and inference from indirect data was demonstrated. A wind velocity field estimation problem highlighted the importance of incorporating statistical constraints for capturing realistic field characteristics from sparse sensors. While both c-LFM and standard LFM accurately captured the mean and variance of the field, the incorporation of the coherence constraint generated significantly more realistic spatial and temporal coherence. Applying the statistical constraint also appeared to help regularize the problem and improve the convergence of the VAE in this case. A material property characterization problem demonstrated that accurate inference of a random field was possible with indirect observations using c-LFM. Apart from providing tractability, the addition of the physical constraint also improved training in the small-data regime. Accuracy of the model when trained with $N = 50$ samples was similar to the case of $N = 100$, representing significant test and cost reduction. The main limitation of the proposed work is that it did not directly address noisy training data, another common challenge in science and engineering. Future work will include extension of c-LFM to imperfect physical or statistical knowledge manifesting as a mismatch between training data and imposed constraints, with noisy data being one such example. Finally, the real-world applications presented will be studied further, where it is expected that the performance could be significantly improved by considering more advanced neural network architectures within the VAE, integrating adaptive training strategies to better balance the VAE loss terms, and carrying out more extensive hyperparameter tuning.

References

Martin S. Alnæs, Jan Blechta, Johan Hake, August Johansson, Benjamin Kehlet, Anders Logg, Chris Richardson, Johannes Ring, Marie E. Rognes, and Garth N. Wells. The FEniCS project version 1.5. *Archive of Numerical Software*, 3(100), 2015. doi: 10.11588/ans.2015.100.20553.

Jan-Hendrik Bastek, WaiChing Sun, and Dennis M. Kochmann. Physics-informed diffusion models, 2025. URL `https://arxiv.org/abs/2403.14404`.

Luigi Carassale and Giovanni Solari. Monte Carlo simulation of wind velocity fields on complex structures. *Journal of Wind Engineering and Industrial Aerodynamics*, 94(5):323–339, 2006. ISSN 0167-6105. doi: https://doi.org/10.1016/j.jweia.2006.01.004. URL `https://www.sciencedirect.com/science/article/pii/S0167610506000079`. The eighth Italian National Conference on Wind Engineering IN-VENTO-2004.

Jacob K Christopher, Stephen Baek, and Ferdinando Fioretto. Constrained synthesis with projected diffusion models, 2024. URL `https://arxiv.org/abs/2402.03559`.

Richard L Coulter and Paytsar Muradyan. Sonic detection and ranging (SODAR) wind profiler instrument handbook. Technical report, DOE Office of Science Atmospheric Radiation Measurement (ARM) Program, 2020.

Quan Dao, Hao Phung, Binh Nguyen, and Anh Tran. Flow matching in latent space. *arXiv preprint arXiv:2307.08698*, 2023.

Agnimitra Dasgupta, Harisankar Ramaswamy, Javier Murgoitio-Esandi, Ken Y. Foo, Runze Li, Qifa Zhou, Brendan F. Kennedy, and Assad A. Oberai. Conditional score-based diffusion models for solving inverse elasticity problems. *Computer Methods in Applied Mechanics and Engineering*, 433:117425, January 2025. ISSN 0045-7825. doi: 10.1016/j.cma.2024.117425. URL `http://dx.doi.org/10.1016/j.cma.2024.117425`.

Arka Daw, M. Maruf, and Anuj Karpatne. PID-GAN: A GAN framework based on a physics-informed discriminator for uncertainty quantification with physics. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery; Data Mining*. ACM, August 2021. doi: 10.1145/3447548.3467449. URL `http://dx.doi.org/10.1145/3447548.3467449`.

Arka Daw, Jie Bu, Sifan Wang, Paris Perdikaris, and Anuj Karpatne. Rethinking the importance of sampling in physics-informed neural networks. *arXiv preprint arXiv:2207.02338*, 2022.

George Deodatis and Michael Shields. The spectral representation method: A framework for simulation of stochastic processes, fields, and waves. *Reliability Engineering & System Safety*, 254:110522, 2025. ISSN 0951-8320. doi: https://doi.org/10.1016/j.ress.2024.110522. URL `https://www.sciencedirect.com/science/article/pii/S0951832024005945`.

Pan Du, Meet Hemant Parikh, Xiantao Fan, Xin-Yang Liu, and Jian-Xun Wang. CoNFiLD: Conditional neural field latent diffusion model generating spatiotemporal turbulence. *arXiv preprint arXiv:2403.05940*, 2024.

Keinosuke Fukunaga and Warren LG Koontz. Application of the Karhunen-Loeve expansion to feature selection and ordering. *IEEE Transactions on computers*, 100(4):311–318, 1970.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

Dominique Guillot, Bala Rajaratnam, and Julien Emile-Geay. Statistical paleoclimate reconstructions via Markov random fields. *The Annals of Applied Statistics*, 9(1), March 2015. ISSN 1932-6157. doi: 10.1214/14-aoas794. URL `http://dx.doi.org/10.1214/14-AOAS794`.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Benjamin Holzschuh and Nils Thuerey. Improving flow matching for posterior inference with physics-based controls. In *ICML 2024 Workshop on Structured Probabilistic Inference & Generative Modeling*, 2024. URL `https://openreview.net/forum?id=9HDL3sH61o`.

Jie Hou, Ying Li, and Shihui Ying. Enhancing pinns for solving pdes via adaptive collocation point movement and adaptive loss weighting. *Nonlinear Dynamics*, 111(16):15233–15261, 2023.

Jiahe Huang, Guandao Yang, Zichen Wang, and Jeong Joon Park. DiffusionPDE: Generative pde-solving under partial observation, 2024. URL `https://arxiv.org/abs/2406.17763`.

Christian Jacobsen, Yilin Zhuang, and Karthik Duraisamy. CoCoGen: Physically-consistent and conditioned score-based generative models for forward and inverse problems, 2024. URL `https://arxiv.org/abs/2312.10527`.

Kari Karhunen. Über lineare methoden in der wahrscheinlichkeitsrechnung. *Ann Acad Sci Fennicae*, 37:1, 1947.

Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. *CoRR*, abs/1312.6114, 2013. URL `https://api.semanticscholar.org/CorpusID:216078090`.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Michel Loeve. Probability theory: foundations, random sequences. *(No Title)*, 1955.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature machine intelligence*, 3(3):218–229, 2021.

R. Mikulevicius and B. L. Rozovskii. Stochastic Navier-Stokes equations for turbulent flows. *SIAM Journal on Mathematical Analysis*, 35(5):1250–1310, 2004. doi: 10.1137/S0036141002409167. URL `https://doi.org/10.1137/S0036141002409167`.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.

Audrey Olivier, Dimitris G Giovanis, BS Aakash, Mohit Chauhan, Lohit Vandanapu, and Michael D Shields. UQpy: A general purpose python package and development environment for uncertainty quantification. *Journal of Computational Science*, 47:101204, 2020.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, et al. GPT-4 Technical Report, 2024. URL `https://arxiv.org/abs/2303.08774`.

Martin Ostoja-Starzewski. Random field models of heterogeneous materials. *International Journal of Solids and Structures*, 35(19):2429–2455, 1998.

M. Raissi, P. Perdikaris, and G.E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2018.10.045. URL `https://www.sciencedirect.com/science/article/pii/S0021999118307125`.

J. N. Reddy. *An Introduction to Continuum Mechanics*. Cambridge University Press, 2 edition, 2013. doi: 10.1017/CBO9781139178952.

Esther Rodrigo-Bonet and Nikos Deligiannis. Physics-guided variational graph autoencoder for air quality inference. In *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6940–6944, 2024. doi: 10.1109/ICASSP48485.2024.10448194.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

Divya Saxena and Jiannong Cao. Generative adversarial networks (GANs) challenges, solutions, and future directions. *ACM Computing Surveys (CSUR)*, 54(3):1–42, 2021.

Tristan A. Shah, Michael C. Stanley, and James E. Warner. Generative modeling of microweather wind velocities for urban air mobility, 2025. URL `https://arxiv.org/abs/2503.02690`.

Masanobu Shinozuka and George Deodatis. Simulation of stochastic processes by spectral representation. 1991.

Alexander Shmakov, Kevin Greif, Michael Fenton, Aishik Ghosh, Pierre Baldi, and Daniel Whiteson. End-to-end latent variational diffusion models for inverse problems in high energy physics. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 65102–65127. Curran Associates, Inc., 2023. URL `https://proceedings.neurips.cc/paper_files/paper/2023/file/cd830afc6208a346e4ec5caf1b08b4b4-Paper-Conference.pdf`.

Dule Shu, Zijie Li, and Amir Barati Farimani. A physics-informed diffusion model for high-fidelity flow field reconstruction. *Journal of Computational Physics*, 478:111972, April 2023. ISSN 0021-9991. doi: 10.1016/j.jcp.2023.111972. URL `http://dx.doi.org/10.1016/j.jcp.2023.111972`.

Bowen Song, Soo Min Kwon, Zecheng Zhang, Xinyu Hu, Qing Qu, and Liyue Shen. Solving inverse problems with latent diffusion models via hard data consistency, 2024. URL `https://arxiv.org/abs/2307.08123`.

Michael A Sutton, Jean Jose Orteu, and Hubert Schreier. *Image correlation for shape, motion and deformation measurements: basic concepts, theory and applications*. Springer Science & Business Media, 2009.

Alexander Tong, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Kilian Fatras, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *arXiv preprint arXiv:2302.00482*, 2023.

Duy Tan Tran, Haakon Robinson, Adil Rasheed, Omer San, Mandar Tabib, and Trond Kvamsdal. GANs enabled super-resolution reconstruction of wind field. In *Journal of Physics: Conference Series*, volume 1669, page 012029. IOP Publishing, 2020.

Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081, 2021.

James E. Warner, Julian Cuevas, Geoffrey F. Bomarito, Patrick E. Leser, and William P. Leser. Inverse estimation of elastic modulus using physics-informed generative adversarial networks, 2020. URL https://arxiv.org/abs/2006.05791.

P. Welch. The use of fast fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms. *IEEE Transactions on Audio and Electroacoustics*, 15(2):70–73, 1967. doi: 10.1109/TAU.1967.1161901.

Jin-Long Wu, Karthik Kashinath, Adrian Albert, Dragos Chirila, Heng Xiao, et al. Enforcing statistical constraints in generative adversarial networks for modeling chaotic dynamical systems. *Journal of Computational Physics*, 406:109209, 2020.

You Xie, Erik Franz, Mengyu Chu, and Nils Thuerey. tempogan: a temporally coherent, volumetric gan for super-resolution fluid flow. *ACM Trans. Graph.*, 37(4), July 2018. ISSN 0730-0301. doi: 10.1145/3197517.3201304. URL https://doi.org/10.1145/3197517.3201304.

Dongbin Xiu. *Numerical methods for stochastic computations: a spectral method approach*. Princeton university press, 2010.

Liu Yang, Dongkun Zhang, and George Em Karniadakis. Physics-informed generative adversarial networks for stochastic differential equations, 2018. URL https://arxiv.org/abs/1811.02033.

Liu Yang, Sean Treichler, Thorsten Kurth, Keno Fischer, David Barajas-Solano, Josh Romero, Valentin Churavy, Alexandre Tartakovsky, Michael Houston, Prabhat, and George Karniadakis. Highly-scalable, physics-informed GANs for learning solutions of stochastic PDEs, 2019. URL https://arxiv.org/abs/1910.13444.

Yibo Yang and Paris Perdikaris. Adversarial uncertainty quantification in physics-informed neural networks. *Journal of Computational Physics*, 394:136?152, October 2019. ISSN 0021-9991. doi: 10.1016/j.jcp.2019.05.027. URL http://dx.doi.org/10.1016/j.jcp.2019.05.027.

Weiheng Zhong and Hadi Meidani. PI-VAE: Physics-informed variational auto-encoder for stochastic differential equations. *Computer Methods in Applied Mechanics and Engineering*, 403:115664, January 2023. ISSN 0045-7825. doi: 10.1016/j.cma.2022.115664. URL http://dx.doi.org/10.1016/j.cma.2022.115664.

A Appendix

A.1 Application Details

A.1.1 Random Field Reconstruction from Sparse Sensors

Additional hyperparameter studies for the random field reconstruction demonstration are illustrated here. Two error metrics are introduced to separate accuracy into two components: mean and covariance. Mean error is here defined as mean squared error, MSE, (over the spatial dimension) of the predicted the mean (in the stochastic dimension). Similarly, the covariance error is defined as the MSE of the predicted covariance function. The mean and covariance are estimated with 1000 generated samples on 100 equally-spaced points for the MSE calculations.

Figure 6 (left) illustrates these two errors as a function of residual weight, λ. As a residual weight increases, a higher penalty is placed upon the statistical residual, R, which results in lower covariance errors. The improvement in covariance error comes at the cost of reduced accuracy in mean predictions; thus, there is an optimal residual weight for this demonstration that balances these two errors (approximately $\lambda = 0.01$). Figure 6 (right) illustrates that errors are relatively invariant to the number of collocation points, C, used to calculate the residual. Using even a very small number of collocation points provides the necessary statistical regularization.

The type of statistical constraint is used for R is compared in Figure 7 as a function of the number of sensors, m, used in training. The comparison is made between a residual based on covariance matching compared to a residual based on correlation matching. The correlation constraint is the same as the covariance constraint without matching the magnitude (i.e. variance). It is seen that both constraints provide benefit over the no-constraint case. However, the correlation constraint, which lacks the magnitude component, must rely on more sensor measurements in order to estimate that portion of the signal.

Figure 8 compares the proposed approach with LFM versus reconstructing the random field by sampling the pre-trained VAE directly. That is, latent vectors are first sampled from the VAE prior distribution, $\mathbf{z}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and then decoded as $u(x) = \mathcal{D}^u_{\boldsymbol{\theta}}(\mathbf{z}, x)$. The accuracy of each approach is compared for estimates of the GP mean function (a) and covariance (b). As expected, using LFM to sample latent representations leads to higher accuracy relative to using the VAE prior. Given the simple nature of the demonstration (and the learned latent space), however, the difference in performance is not substantial.

A.1.2 Random Field Inference from Limited Observations

Additional hyperparameter studies for the random field inference demonstration are illustrated here. As in the previous section, the hyperparameters of residual weight, λ, number of collocation points, C, and number of data points, N, are investigated. Figure 9 illustrates that there is relative insensitivity to these parameters with a couple notable exceptions. First, performance degrades at large residual weights ($\lambda_p \geq 1.0$). And second, there is a consistent trend of more accurate results as the number of data points increases. The mean and variances are estimated with 1000 generated samples on 100 equally-spaced points for the MSE calculations.

Figure 10 compares the proposed approach with LFM versus reconstructing the random field by sampling the pre-trained VAE directly. The accuracy of inference of the unknown random field, $V(x)$, is shown in terms of mean in Figure 10 (a) and variance in Figure 10 (b). Sampling with LFM leads to substantially higher accuracy for all training dataset sizes considered.

A.1.3 Wind Velocity Field Estimation

Further information on the random wind velocity field reconstruction application are provided here, focusing in particular on details of the process for generating training data, the implementation of the statistical coherence constraint residual, and additional results that further illustrate the performance of the proposed approach.

Training Data Generation

Wind velocity training data was generated based on the classical spectral representation approach Shinozuka and Deodatis [1991] for simulating stochastic processes, using a description of spectral properties borrowed from wind engineering literature [Carassale and Solari, 2006]. Here, the formulation was motivated by the need to model wind loading on large structures like bridges probabilistically, so the statistical properties of the training data used here can be considered realistic in this context.

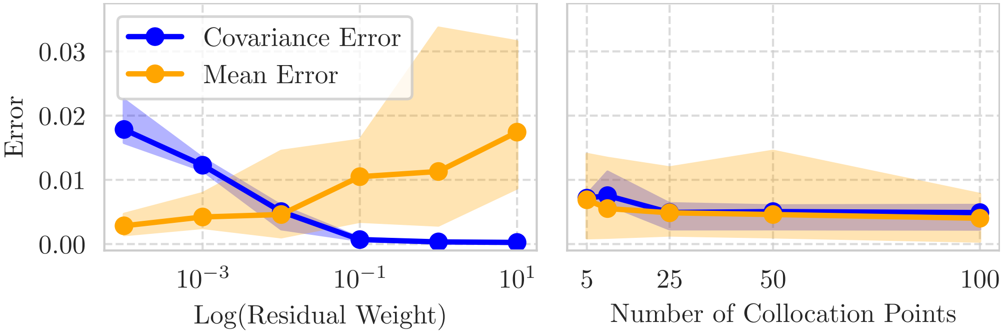


Figure 6: Errors of random field reconstruction when using a covariance constraint. Here the number of sensors is $m = 3$ and (left) the number of collocation points $C = 50$, (right) the residual weight is $\lambda = 0.01$

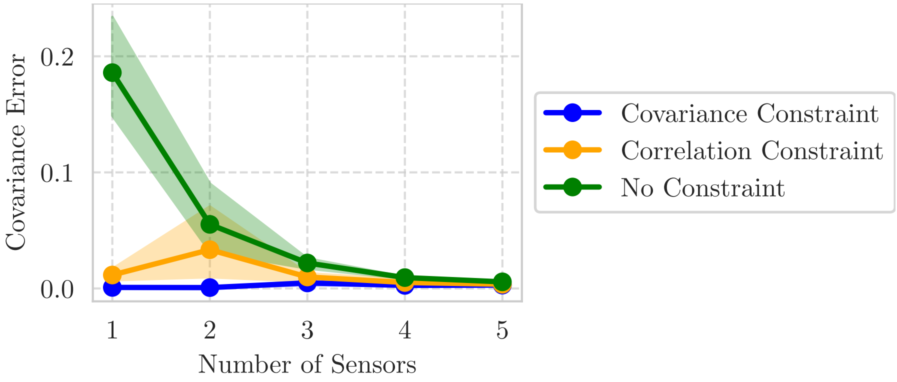
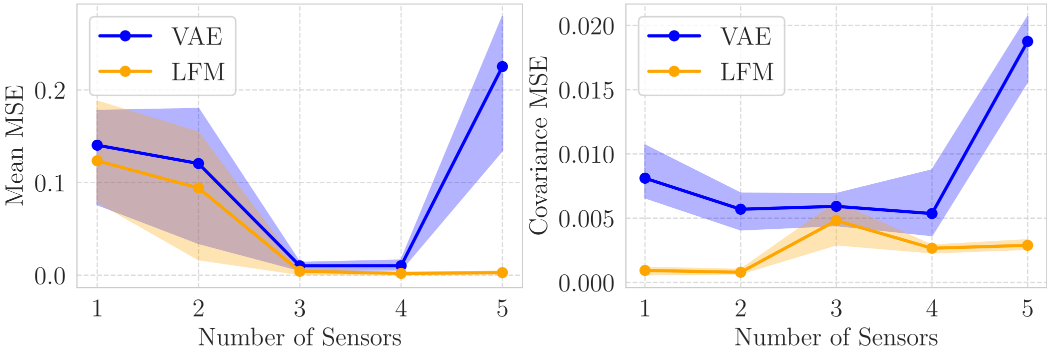


Figure 7: Comparison of constraint types on errors of random field reconstruction. Here the number of collocation points is $C = 50$ and the residual weight is $\lambda = 0.01$.



(a)

(b)

Figure 8: Random field reconstruction using LFM versus the pre-trained VAE directly, comparing the MSE in the a) mean and b) covariance estimates.

As mentioned in Section 5.3, wind velocity is idealized as being comprised of two components via Equation 9: a deterministic mean function,

$$\boldsymbol{\mu}_u(\mathbf{x}) = 2.5u_*\text{ln}\left(\frac{x_3}{z_0}\right), \tag{11}$$

with shear velocity, $u_* = 1.8$ m/s, and surface roughness, $z_0 = 0.015$ m, and random turbulence component, $\mathbf{W}(\mathbf{x}, \omega)$, modeled as a zero-mean Gaussian process. The main task in obtaining training data for $\mathbf{U}(\mathbf{x}, \omega)$ is thus to generate random samples of $\mathbf{W}$ with prescribed statistical properties. Here, we largely follow Carassale and Solari [2006] for the formulation of these properties for realistic wind fields, with the simplification of focusing on a single component of velocity, U_1, and assuming $\mathbf{W}$ is aligned with axes of the global reference system, removing the need for coordinate transformations.

The basis of the formulation is the definition of the cross-power spectral density (CPSD) function (the generalization of the covariance function to include a dependence on frequency, n) describing W,

$$S_{11}^{CP}(\mathbf{x}, \mathbf{x}', n) = \sqrt{S_1(\mathbf{x}; n)S_1(\mathbf{x}'; n)}\text{Coh}(\mathbf{x}, \mathbf{x}', n) \tag{12}$$

where S_{11} is the auto spectral density of the W_1 turbulence component and Coh is the coherence function, as defined in Equation 10, using decay coefficients $c = [3, 3, 0.5]$. A suitable definition of S_{11} for modeling turbulence is given as follows

$$\frac{nS_{11}(\mathbf{x}; n)}{\sigma_1^2(\mathbf{x})} = \frac{\lambda_1 n(L_1/\|\boldsymbol{\mu}_u(\mathbf{x})\|)}{(1 + 1.5\lambda_1 n(L_1/\|\boldsymbol{\mu}_u(\mathbf{x})\|))^{5/3}} \tag{13}$$

where $\lambda_1 = 6.868$ and σ_1^2 and L_1 are the variance and integral length scale of W_1, given by

$$\begin{aligned} \sigma_1^2 &= [6 - 1.1\text{atan}(\text{ln}z_0 + 1.75)]u_*^2, \\ L_1 &= 300(\frac{x_3}{200})^{0.67+0.05\text{ln}z_0}. \end{aligned} \tag{14}$$

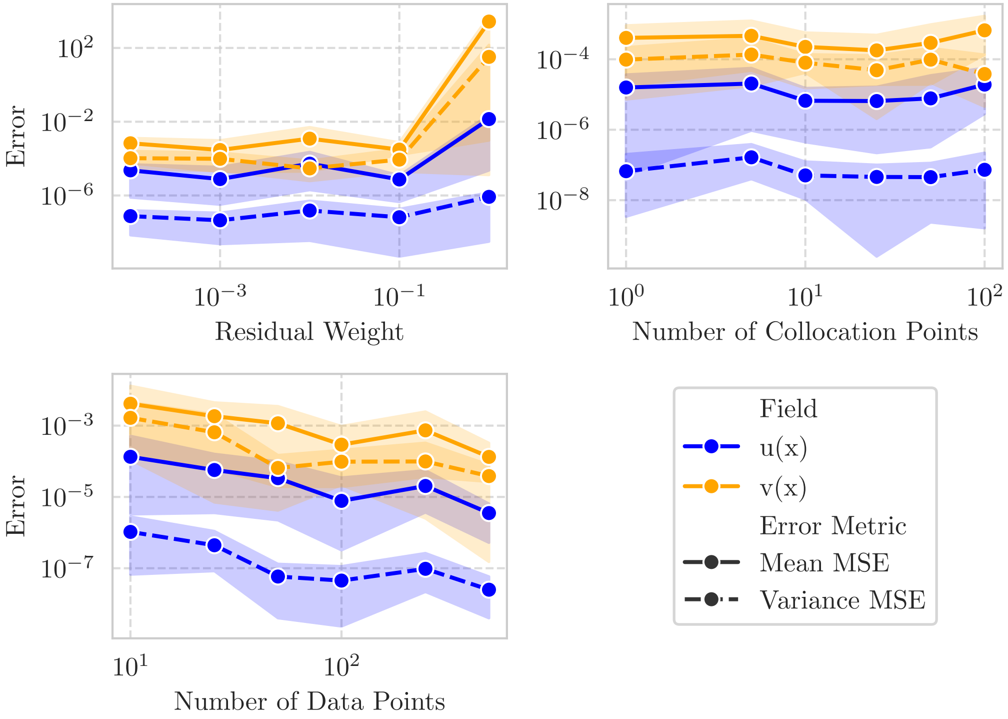
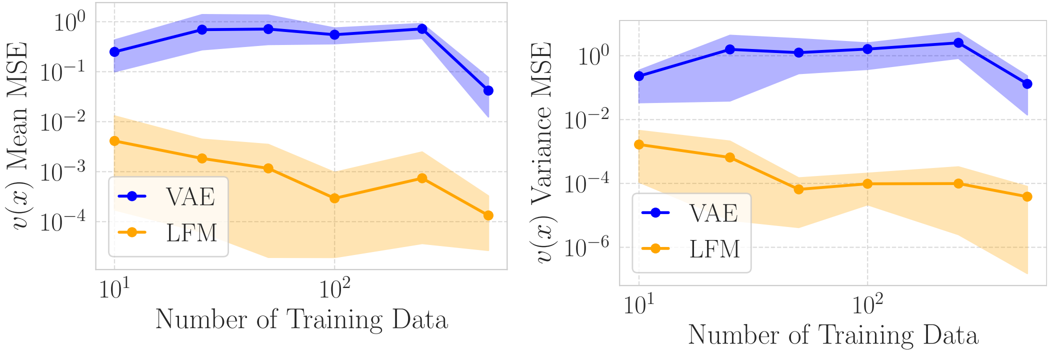


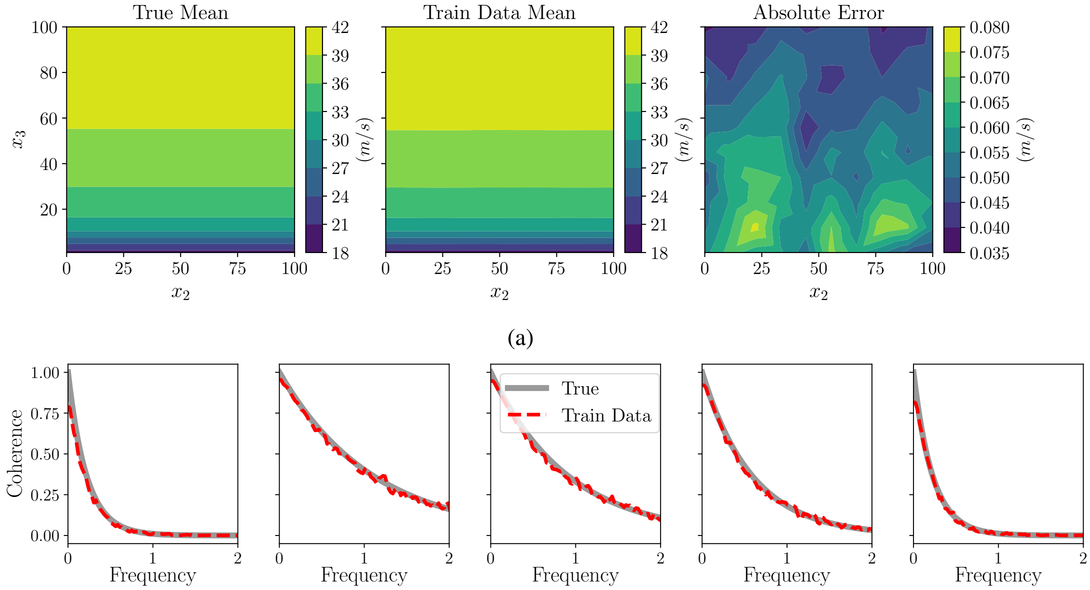
Figure 9: Effect of hyperparameters on random field inference demonstration. Here the number of sensors is $m = 25$, the number of data points is $N = 100$, the number of collocation points is $C = 50$, and the residual weight is $\lambda = 0.001$, except where the x-axes indicate variation.



(a)

(b)

Figure 10: Random field inference using LFM versus the pre-trained VAE directly, comparing the MSE in the a) mean and b) variance of the inferred random field $V(x)$.



(a)

(b)

Figure 11: Wind velocity field training data verification. Comparison of the true and empirically-calculated values from the training data for the (a) mean wind field and (b) coherence between the wind velocity at five random pairs of spatial coordinates.

The spectral representation method (SRM) was employed to generate samples of W_1 using an open-source implementation in `UQpy` [Olivier et al., 2020]. SRM is a classical framework for simulating random processes and fields that uses the fact that a zero-mean, stationary stochastic process with known CPSD can be approximated by a finite series representation of cosine terms with random phase angles [Deodatis and Shields, 2025]. To apply the SRM, a CPSD matrix, $\mathbf{S}^{CP} \in \mathbf{R}^{N_{x_2} \times N_{x_3} \times N_f}$, must be assembled by evaluating the CPSD in Equation 12 on a spatial and frequency discretization. A 10×10 uniformly spatial grid across $x_2 \in [0, 100]$ m and $x_3 \in [H_{min}, 100]$ m was used, where $H_{min} = 1$ to avoid numerical issues due to Equation 11 for $x_3 \leq z_0$. A grid of $N_f = 128$ uniform points for $f \in [0, 3.0]$ Hz was used to discretize frequency ($\Delta_f = 3.0/N_f$). Following Carassale and Solari [2006], this choice of frequency discretization simulates $T = 1/\Delta_f = 42.5$ seconds with $N_t = 2 \times N_f = 256$ time steps. Sparse measurement data was simulated for the example by keeping every third vertical column of sensors, starting at $x_2 = 0$ m, filtering data from 60 total grid points (see Figure 4 a). A Python implementation of the above procedure is included in the code repository for the paper.

A verification of synthetically-generated wind velocity used for training data is shown in Figure 11. Here, figure (a) compares the mean of the training data to the true mean function in Equation 11 and (b) compares the empirically-calculated coherence using the training data (see next section) with the prescribed coherence in Equation 10. It can be seen that good agreement is observed between the training data and prescribed values.

Residual Evaluation

The wind field reconstruction example leverages a statistical constraint that relies on the residual between the true coherence function (Equation 10) and the empirically estimated value from DGM wind velocity samples, $\hat{u}(\mathbf{x})$. Let the generated wind velocity at a given spatial collocation point, $\tilde{\mathbf{x}}^{(c)} = (\tilde{x}_2^{(c)}, \tilde{x}_3^{(c)})$, be given as $\hat{u}_1^c \equiv \hat{u}(\tilde{x}_2^{(c)}, \tilde{x}_3^{(c)}, t)$. The coherence between wind velocities at two different collocation points, $\tilde{\mathbf{x}}^{(c_1)}$ and $\tilde{\mathbf{x}}^{(c_2)}$, can be empirically estimated as:

$$\gamma^2(f) = \frac{|P_{12}(f)|^2}{P_{11}(f)P_{22}(f)}, \tag{15}$$

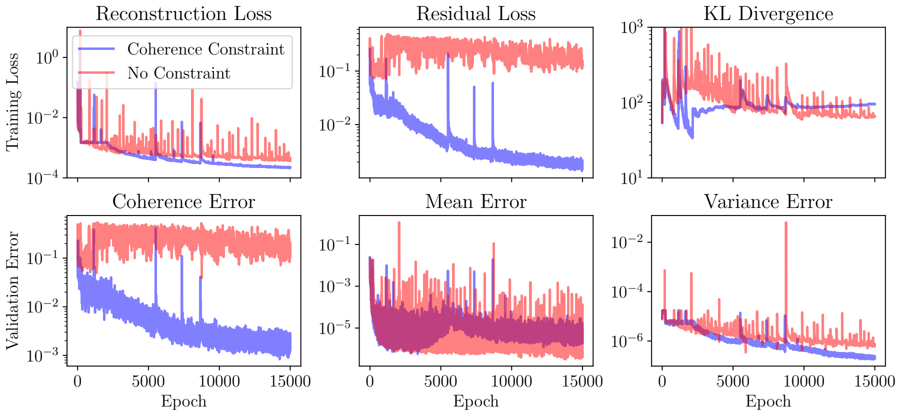


Figure 12: VAE convergence for the wind velocity estimation application. The loss function terms (top) and validation metrics (bottom) versus epoch both with and without the coherence constraint.

where P_{12} is the (empirical) CPSD between $\hat{u}_1^{c_1}$ and $\hat{u}_1^{c_2}$ and P_{11} and P_{22} are the (empirical) auto spectra of $\hat{u}_1^{c_1}$ and $\hat{u}_1^{c_2}$, respectively. The empirical spectral densities, P_{**}, are computed using Welch's method [Welch, 1967] based on the Fast Fourier Transform (FFT) in PyTorch. Finally, the coherence residual for a single pair of collocation points is calculated as follows:

$$\frac{1}{N_f}\sum_{i=1}^{N_f} \|\gamma^2(f^{(i)}) - \text{Coh}(\tilde{\mathbf{x}}^{(c_1)}, \tilde{\mathbf{x}}^{(c_2)}, f^{(i)})\|^2. \tag{16}$$

Additional Results

Figure 12 compares the convergence of LFM both with and without the coherence constraint. The addition of the coherence constraint appears to add some stability to the training process, as evidenced by the periodic spikes in the loss function for the conventional LFM that are not present with the constraint. Both approaches achieve similar error in the mean and variance estimates on the validation set, but the conventional approach cannot match the coherence of the true data.

Figure 13 compares a representative wind velocity sample from the test dataset versus generated samples both with and without the coherence constraint incorporated during training. Four snapshots in time are shown for each sample. It can be seen that imposing the coherence constraint with c-LFM allows more realistic variability in the horizontal direction to be captured, while the generated sample with no constraint has nearly no variation in the horizontal direction due to the sparseness of the sensors in this direction (only four locations along x_2 - see Figure 4 (a)).

Table 1 compares the errors in estimated wind velocity statistics from sampling with LFM versus sampling the pre-trained VAE directly both with and without the coherence constraint. While the errors are comparable for the coherence and mean, the variance MSE is an order of magnitude higher when sampling the VAE compared to using LFM. The statistics used for the error calculations were estimated from 1000 samples for both the generated and test data.

Table 1: Errors in wind statistics for sampling with LFM versus the pre-trained VAE both with and without the coherence constraint.

	Sampling	**Coherence MSE**	**Mean MSE**	**Variance MSE**
With Constraint	LFM	8.25e-03	1.00e-01	1.61e+00
	VAE	1.28e-02	3.41e-02	1.46e+01
No Constraint	LFM	1.02e-01	1.53e-01	2.10e+00
	VAE	9.23e-02	1.78e-02	4.25e+01

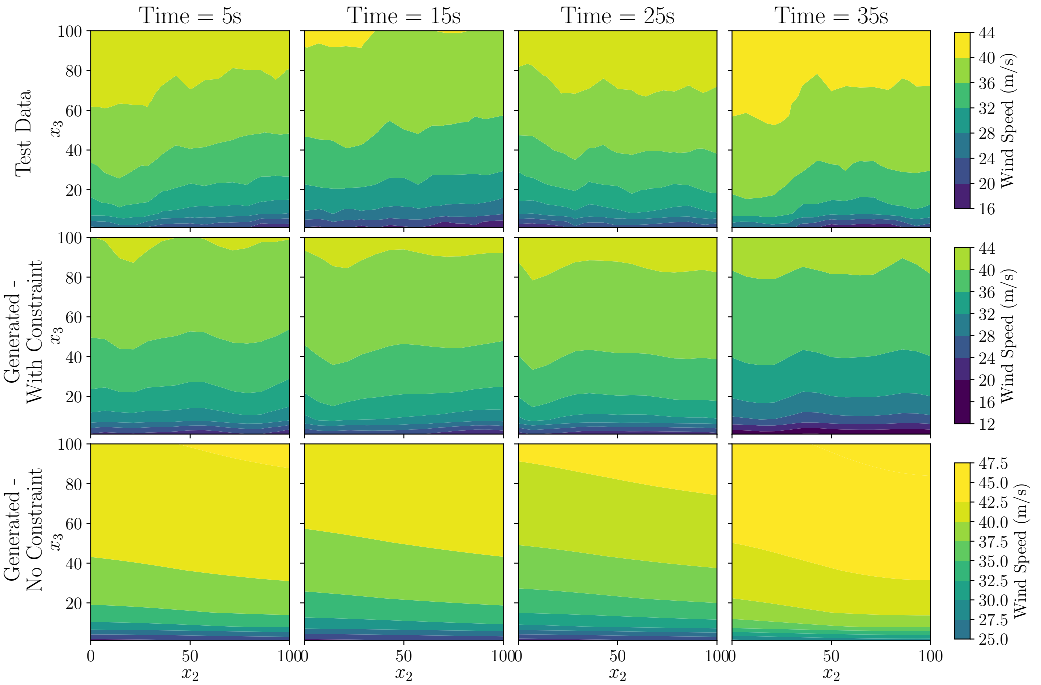


Figure 13: Representative wind velocity samples at four different instances in time for the test data (top), generated with coherence constraint (middle), and generated without coherence constraint (bottom).

A.1.4 Material Property Characterization

Additional details regarding the material property inference problem are included here to aid reproducibility. The governing PDE is given explicitly as:

$$\mathcal{N}_x(\mathbf{u}, \upsilon) = \nabla \cdot \boldsymbol{\sigma}(\mathbf{u}, \upsilon) = \mathbf{0}. \tag{17}$$

where $\boldsymbol{\sigma}$ is the second order stress tensor. Assuming a two-dimensional plane-stress formulation, isotropic elasticity, and small strain theory Reddy [2013], the stress tensor is

$$\boldsymbol{\sigma} = \frac{2V}{1+\upsilon}\left[\nabla\mathbf{u} + \nabla\mathbf{u}^T + \frac{2\upsilon}{1-\upsilon}\mathrm{Tr}(\nabla\mathbf{u})\mathbf{I}\right] \tag{18}$$

where V is the spatially varying elastic modulus (stiffness), υ is the Poisson ratio, and $\mathbf{I}$ is the identity tensor. We assumed a constant $\upsilon = 0.3$. The boundary conditions are $u_1 = 0$ for $\mathbf{x} \in \Gamma_1$, $u_2 = 0$ for $\mathbf{x} = (0, 0)$ and

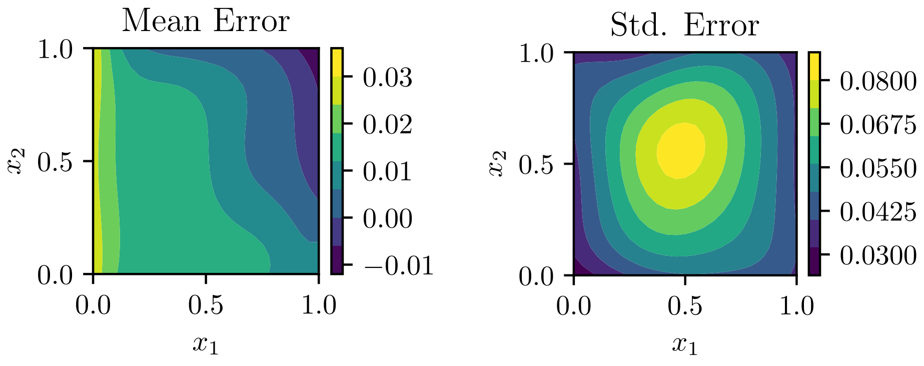
$$\boldsymbol{\sigma}(\mathbf{u}, V) \cdot \mathbf{n} = \boldsymbol{\tau} \tag{19}$$

where $\mathbf{n}$ is the boundary outward normal, $\boldsymbol{\tau} = [0, 0]$ on Γ_2 and Γ_3 and $\boldsymbol{\tau} = [1.5, 0]$ on Γ_4.

In this application, the random field $V(\mathbf{x}, \omega)$ represents the aleatory uncertainty associated with a family of test articles with spatial and test-to-test variability. Relatively dense measurements are feasible in real-world applications through the use of digital image correlation [Sutton et al., 2009], but the total number of test articles is often $10^1 - 10^2$. The ground truth material property was modeled as the lognormal random field, implemented by transforming a zero-mean Gaussian process using an exponential function,

$$V(\mathbf{x}, \omega) = \alpha + \beta \exp(g(\mathbf{x}, \omega)), \tag{20}$$

$$g(\mathbf{x}, \omega) \sim \mathcal{GP}\left(0, \exp(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2l^2})\right), \tag{21}$$



(a)

(b)

Figure 14: The absolute error between the generated and true (a) mean and (b) standard deviation fields in the inferred random field, $V(x, \omega)$ for $N = 1000$.

with $\alpha = 1.0$, $\beta = 0.1$, and correlation length $l = 1.0$ [Warner et al., 2020]. Samples were generated by computing a truncated Karhunen-Loeve (KL) expansion Fukunaga and Koontz [1970] for $g(\mathbf{x})$ and then applying the transformation in Equation 20. A five-term KL expansion was used, retaining $\sim 99\%$ of the variance of $g(\mathbf{x})$. Using the boundary conditions described in Section 5.4, the training data was then generated using a forward solve of a finite element model implemented in the `FEniCS` Python library [Alnæs et al., 2015] with the resulting $\hat{\mathbf{u}}$ fields discretized as described in Section 5.4 per simulated test specimen.

Comparisons of the estimated PDFs and spatial correlations are shown in Section 5.4. Additional results are shown in Figure 14, showing the absolute errors in the estimated mean and standard deviation fields of the inferred material property field estimated from 1000 samples. These results help to put into context the mean and standard deviation L2 errors versus N that are shown in Figure 5(d).

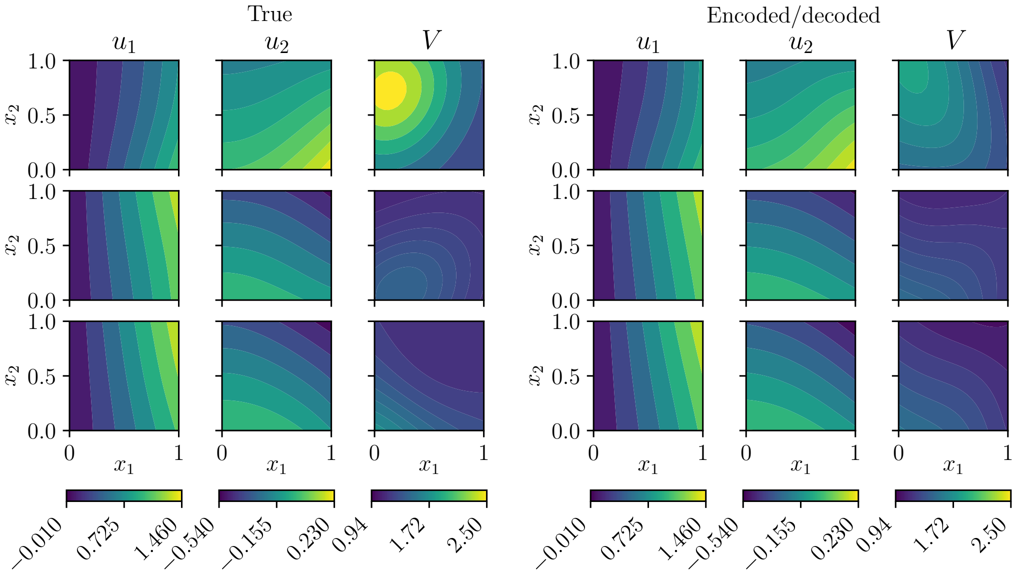
A qualitative assessment of the VAE is presented in Figure 15. Three ground truth samples are shown in subfigure (a), with the corresponding encoded/decoded ($N = 1000$) fields shown in subfigure (b). The encoder and decoder must work in concert to correctly recreate the sampled fields, including correct application of the physical constraints since field samples $V(\mathbf{x}, \hat{\omega})$ are unknown and not provided at training time nor to the encoder in this example. While there is close correspondence between the true and encoded/decoded observations (u_1 and u_2), there is noticeable discrepancy between samples of the inferred field, V, indicating a relative lack of accuracy in the enforcement of the physical constraint. Given the well-known challenges in training neural networks with physics residuals [Wang et al., 2021], more advanced training approaches, including adaptive loss weighting [Hou et al., 2023] and collocation point sampling [Daw et al., 2022], could improve the performance when using c-LFM with physical constraints.

Once trained, LFM allows for the generation of new samples, as illustrated in Figure 16. Note that these samples were randomly generated and do not correspond with the samples shown in Figure 15.

Figure 17 compares the material property inference accuracy when generating samples with LFM versus using the pre-trained VAE directly, where (a) shows the relative L2 error in the estimated mean field and (b) shows the same error metric for the variance field. LFM typically provides better accuracy for different training dataset sizes, but in many cases the errors are comparable. This indicates that there is potentially too much regularization in the training of the VAE which is causing limited expressivity in the latent space representation of the random fields considered. Thus, increasing the capacity of the encoder and function decoder could improve the results and will be pursued in future work.

A.2 Implementation Details

This section provides further implementation details for the proposed approach, including concrete algorithms for training and sampling and a description of hyperparameters used for the experiments.



(a)

(b)

Figure 15: (a) Samples from the true **U** and V random fields and (b) encoded/decoded versions ($N = 1000$) of the same samples.

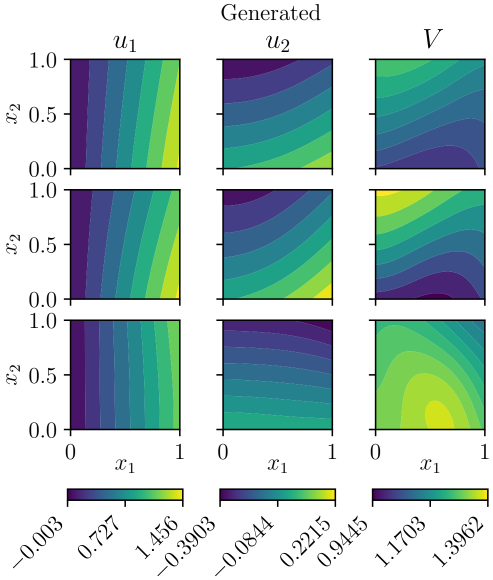
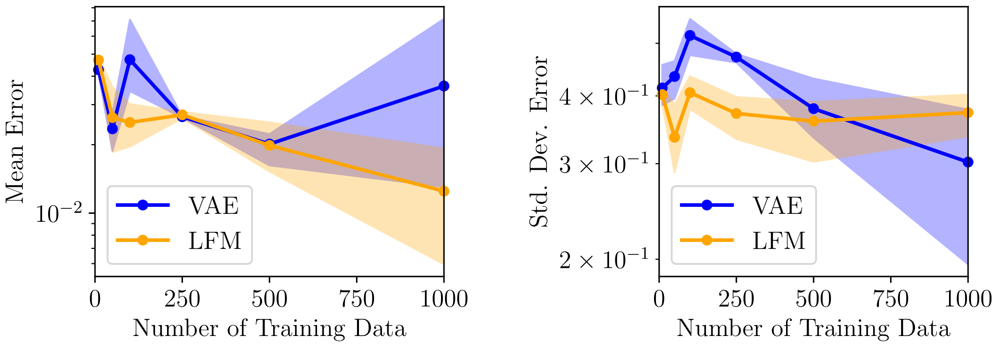


Figure 16: Generated samples from the estimated **U** and V random fields ($N = 1000$).



(a)

(b)

Figure 17: Inference of the unknown material property field using LFM versus the pre-trained VAE directly, comparing the relative L2 error in the estimated a) mean and b) variance.

The main training pipeline for the VAE was implemented using PyTorch Lightning with GPU acceleration and model checkpointing based on the minimum training loss. The Adam optimizer was used with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and learning rates provided in Appendix A.2.3. All experiments were conducted on an internal computing node equipped with eight NVIDIA A100 GPUs (80GB memory per GPU), 128 CPU cores, and 750GB of system memory.

A.2.1 Algorithms

Algorithms 1 and 2 summarize the training procedure for the VAE encoder/decoder networks and the latent flow matching velocity network, respectively. Sampling using the trained velocity network is summarized in Algorithm 3. Details regarding the evaluation of the constraint loss terms in Algorithm 1 can be found in Appendix A.2.2.

Algorithm 1 Training VAE with Function Decoder and Residual Constraints

```
Require: Encoder $\mathcal{E}_\phi$, DeepONet decoder $\mathcal{D}_\theta$, weights $\lambda_{kl}$, $\lambda_r$, $\lambda_f$
for each training iteration do
    Sample batch $\{\mathbf{y}^{(j)}\}_{j=1}^{B}$ and collocation points $\{\tilde{\mathbf{x}}^{(c)}\}_{c=1}^{C}$
    for each $j$ in batch do
        $\mu_\phi, \sigma_\phi = \mathcal{E}_\phi(\mathbf{y}^{(j)})$
        $\mathbf{z}^{(j)} = \mu_\phi + \epsilon \circ \sigma_\phi$, with $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
        Compute reconstruction: $\hat{\mathbf{y}}^{(j)} = \mathcal{T}(\mathcal{D}_\theta^u(\mathbf{z}^{(j)}, \mathbf{x}))$
        $\mathcal{L}_{rec} = \|\hat{\mathbf{y}}^{(j)} - \mathbf{y}^{(j)}\|^2$
        $\mathcal{L}_{kl} = D_{KL}(q_\phi(\mathbf{z}^{(j)}|\mathbf{y}^{(j)}), p(\mathbf{z}))$
        $\mathcal{L}_{stat} = \|R(\mathcal{D}_\theta^u(\mathbf{z}^{(j)}, \tilde{\mathbf{x}}^{(c)}))\|^2$
        $\mathcal{L}_{phys} = \|F(\mathcal{D}_\theta^u(\mathbf{z}^{(j)}, \tilde{\mathbf{x}}^{(c)}), \mathcal{D}_\theta^v(\mathbf{z}^{(j)}, \tilde{\mathbf{x}}^{(c)}))\|^2$
    end for
    $\mathcal{L} = \mathcal{L}_{rec} + \lambda_{kl}\mathcal{L}_{kl} + \lambda_r\mathcal{L}_{stat} + \lambda_f\mathcal{L}_{phys}$
    Update parameters $\phi, \theta$ by minimizing $\mathcal{L}$
end for
Ensure: Trained encoder $\mathcal{E}_\phi$ and decoder $\mathcal{D}_\theta$
```

Algorithm 2 Latent Flow Matching Training

```
Require: Trained encoder $\mathcal{E}_\phi$, velocity network $\boldsymbol{\nu}_\eta$
for each training iteration do
    Sample $\{\mathbf{y}^{(j)}\}_{j=1}^{B} \sim p(\mathbf{y})$
    $\mathbf{z}_0^{(j)} = \mathcal{E}_\phi(\mathbf{y}^{(j)})$, $\mathbf{z}_1^{(j)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
    Sample $t^{(j)} \sim \text{Uniform}(0, 1)$
    $\mathbf{z}_t^{(j)} = (1 - t^{(j)})\mathbf{z}_0^{(j)} + t^{(j)}\mathbf{z}_1^{(j)}$
    $\boldsymbol{\nu}^{(j)} = \mathbf{z}_1^{(j)} - \mathbf{z}_0^{(j)}$
    Minimize $\mathcal{L} = \frac{1}{B}\sum_{j=1}^{B} \|\boldsymbol{\nu}^{(j)} - \boldsymbol{\nu}_\eta(\mathbf{z}_t^{(j)}, t^{(j)})\|_2^2$
end for
Ensure: Trained velocity network $\boldsymbol{\nu}_\eta$
```

Algorithm 3 Latent Flow Matching Sampling

```
Require: Trained velocity network $\boldsymbol{\nu}_{\eta^*}$, trained decoder $\mathcal{D}_\theta^u$
$\mathbf{z}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
Solve $\frac{d\mathbf{z}_t}{dt} = -\boldsymbol{\nu}_{\eta^*}(\mathbf{z}_t, t)$ from $t = 1$ to $t = 0$, starting from $\mathbf{z}_1$
$\mathbf{u}(\mathbf{x}) = \mathcal{D}_\theta^u(\mathbf{z}_0, \mathbf{x}) = \sum_{k=1}^{p} b_k^u(\mathbf{z}_0; \psi) t_k^u(\mathbf{x}; \xi)$
Ensure: Continuous function $\mathbf{u}(\mathbf{x})$
```

A.2.2 Evaluating Constraint Residuals

The manner in which the residuals are computed varies between statistical and physical constraints. For the former, any statistics appearing in R must be estimated using the current batch of samples during training,

$$\begin{aligned} &\mathbb{E}_{\mathbf{z}\sim\mathcal{E}_\phi(\mathbf{y}^{(j)})}\left[R(U(\mathbf{x}, \mathbf{z}(\omega))\right] = 0 \\ &\approx \frac{1}{N_B}\sum_{i=1}^{N_B}\left[R(\mathcal{D}_\theta^u(\mathbf{z}^{(j,i)}, \mathbf{x}))\right] \end{aligned} \tag{22}$$

where $N_B = \frac{N}{B}$ is the number of samples per batch and $\mathbf{z}^{(j,i)}$ is the i^{th} latent variable sample in batch j. Thus, the residual of the statistical constraint is evaluated at the set of collocation points as follows:

$$\|R(\mathcal{D}_\theta^u(\mathbf{z}^{(j)}, \mathbf{x}))\|^2 \approx \frac{1}{C}\sum_{c=1}^{C}\left\|\frac{1}{N_B}\sum_{i=1}^{N_B}\left[R(\mathcal{D}_\theta^u(\mathbf{z}^{(j,i)}, \tilde{\mathbf{x}}^{(c)}))\right]\right\|^2. \tag{23}$$

On the other hand, the physical constraints are intended to hold both $\forall\mathbf{x}$ and for all samples of $\mathbf{u}$ and $\mathbf{v}$. Thus, the residuals are evaluated on a per-latent-variable-sample basis as follows:

$$\|F(\mathcal{D}_\theta^u(\mathbf{z}^{(j)}, \mathbf{x}), \mathcal{D}_\theta^v(\mathbf{z}^{(j)}, \mathbf{x}))\|^2 \approx \frac{1}{C}\frac{1}{N_B}\sum_{c=1}^{C}\sum_{i=1}^{N_B}\left|F(\mathcal{D}_\theta^u(\mathbf{z}^{(j,i)}, \tilde{\mathbf{x}}^{(c)}), \mathcal{D}_\theta^v(\mathbf{z}^{(j,i)}, \tilde{\mathbf{x}}^{(c)})))\right|^2 \tag{24}$$

A.2.3 Network Architecture and Hyperparameters

Network architecture and hyper-parameters used for each of the four examples are summarized in Table 2. Most of the networks used were multi-layer perceptrons (MLPs) with GELU activation functions. In the two application examples, it was found that more expressive architectures for the branch and encoder was beneficial for the added problem complexity. For these examples, an enhanced branch network was used with residual blocks containing SiLU activations, layer normalization, and skip connections to improve gradient flow while enabling deeper representations necessary for capturing the more complex function spaces (the number of hidden layers in Table 2 corresponds to number of residual blocks used). For the wind velocity field estimation example, the encoder portion of the U-Net architecture[Ronneberger et al., 2015] was adopted for a more capable encoder.

Table 2: Neural network architecture and training details

Parameter	Reconstruction Demo	Inference Demo	Wind Estimation	Material Characterization
Batch Size (per GPU)	256	256	128	256
Learning Rate	10^{-3}	10^{-3}	5×10^{-4}	10^{-3}
Latent Dimension	4	4	128	64
DeepONet dimension (p)	64	64	128	128
Encoder # hidden layers	3	3	UNet	3
Branch # hidden layers	2	2	2	2
Trunk # hidden layers	2	2	3	3
Encoder hidden layer width	128	128	UNet	128
Branch hidden layer width	128	128	128	128
Trunk hidden layer width	128	128	128	128
λ_{kl}	10^{-6}	10^{-6}	10^{-7}	10^{-6}
λ_r	10^{-3}	–	10^{-2}	–
λ_f	–	10^{-2}	–	10^{-6}
Epochs	10k	10k	15k	20k
# GPUs	1	1	4	1
Approx. Training Time	10 min	15 min	40 hours	6 hours

For the flow matching implementation, a MLP with three hidden layers of 128 units each and GELU activation functions was used for the flow model for all examples. Training was performed using the Adam optimizer with a learning rate of 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and a batch size of 128 for 1000 epochs. During training, a small noise component (standard deviation of 0.01) was added to the interpolated points to improve stability. For sampling, we leveraged the `Torchdyn` library's neural differential equation implementation with an adjoint-based 4th-order Runge-Kutta (RK4) solver to integrate the learned vector field with 100 evenly spaced time steps between $t = 0$ and $t = 1$. Very little hyperparameter tuning was performed for flow matching in this study, so it is expected that some accuracy improvements are possible for the results shown herein.

A.2.4 Code Attribution

We acknowledge the adoption of portions of the following open-source resources for the development of code for this work:

The Annotated Diffusion Model
Authors: Niels Rogge and Kashif Rasul
Source: `https://huggingface.co/blog/annotated-diffusion`
Components Used: U-Net architecture implementation
License: N/A
Notes: We adapted the U-Net architecture with modifications to suit our specific requirements for function decoders.

Conditional Flow Matching
Authors: Alexander Tong and Kilian Fatras
Source: `https://github.com/atong01/conditional-flow-matching`
Components Used: PyTorch wrapper for integrating flow matching with TorchDyn
License: MIT
Notes: The wrapper was used to interface our flow model with the Neural ODE solver for sampling from the learned flow.

A.3 Broader Impacts

The proposed work is motivated by allowing breakthroughs in DGM research to be applied successfully for scientific discovery and engineering applications where massive training datasets may not be available. While incorporating prior knowledge and constraints into DGMs may have more general utility beyond the applications considered herein, we expect the positive societal impact of our work to primarily lie in scientific domain. As with any approach that considers prior knowledge as an input (e.g., Bayesian inference), a potential negative impact is the biasing of results when this information is (knowingly or unknowingly) incorrect. Thus, care must be taken when formulating physical/statistical constraints used to supplement training data with the proposed approach, and the assumptions being made should be clearly articulated.