Stochastic Process Learning via Operator Flow Matching

Yaozhong Shi
California Institute of Technology
yshi5@caltech.edu

Zachary E. Ross
California Institute of Technology
zross@caltech.edu

Domniki Asimaki
California Institute of Technology
domniki@caltech.edu

Kamyar Azizzadenesheli
NVIDIA Corporation
kaazizzad@gmail.com

Abstract

Expanding on neural operators, we propose a novel framework for stochastic process learning across arbitrary domains. In particular, we develop operator flow matching (OFM) for learning stochastic process priors on function spaces. OFM provides the probability density of the values of any collection of points and enables mathematically tractable functional regression at new points with mean and density estimation. Our method outperforms state-of-the-art models in stochastic process learning, functional regression, and prior learning.

1 Introduction

Stochastic processes are foundational to many domains, from functional regression and physics-based data assimilation, to financial markets, geophysics, and black box optimization. Stochastic processes can serve as prior distributions over functions and can provide the density of any finite collection of points. Conventionally, these priors are designed by hand from predefined Gaussian processes (GP) and their variants, and therefore assume that they adequately describe the phenomena of interest. However, many processes modeled in the natural world are not well described by GP, Fig 2. Such models limit the flexibility and generalizability of these stochastic processes in real-world applications, leaving behind significant challenges for more general stochastic process learning (SPL).

In SPL, the prior over the stochastic process is learned from data, i.e., a set of historical point evaluations in past experiments. Learning the prior over the process is crucial for universal functional regression (UFR), which is a recently proposed Bayesian scheme for functional regression and takes GP-regression as a special case when the prior is Gaussian [1]. UFR is important to scientific and engineering domains, including reanalysis, data completion-assimilation, uncertainty quantification, and black box optimization.

In this paper, we introduce a novel operator learning framework termed operator flow matching (OFM) for learning priors over stochastic processes through the joint distribution of any collection of points. To achieve this, we theoretically and empirically generalize marginal optimal transport flow matching [2] to infinite-dimensional function spaces where we map a GP into a prior over function spaces through a flow differential equation. We then derive SPL from the function space derivation and learn a prior over arbitrary sets of points. For SPL, we map any collection of pointwise evaluations of a GP to pointwise evaluations of target functions. This allows us to learn prior distributions over more general stochastic processes, hence enabling sampling values of any collection of points with their associated density and facilitating efficient UFR. We leverage this capability by extending neural operators [3]–designed initially to map functions between infinite-dimensional spaces–to maps

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

between collections of points deploying their functional convergence properties. This serves as the essential architecture block in OFM.

After learning the prior and having access to the densities, OFM can be used for UFR, where given any collection of points of the underlying function, we estimate the posterior mean value of any new collection of points and efficiently sample from their posterior values using stochastic gradient Langevin dynamics (SGLD) [4], i.e., a Gaussian sampling on the input GP space (Fig 1). We show that OFM significantly outperforms state-of-the-art (SOTA) methods, including deep GPs, conditional models, and operator flows (OPFLOW) [1, 5–10].

Generalizing GP-regression to regression over general stochastic process in a practical and implementable way demands a unified framework that spans many key fields. Because readers will have diverse backgrounds and expertise, we provide a set of potential questions and answers in Appendix A to further clarify the development and highlight our contributions.

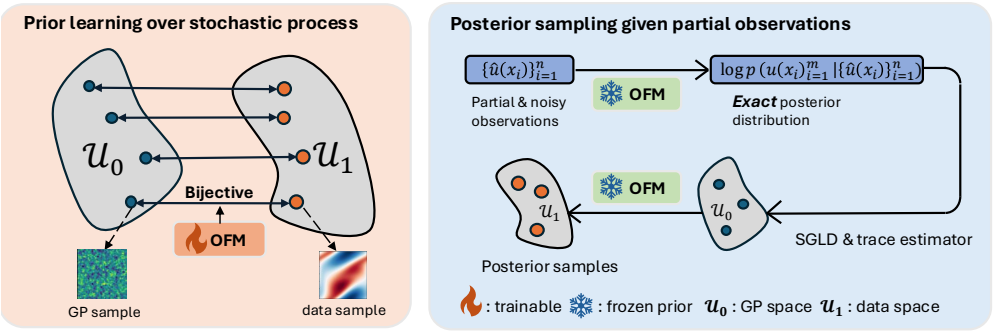


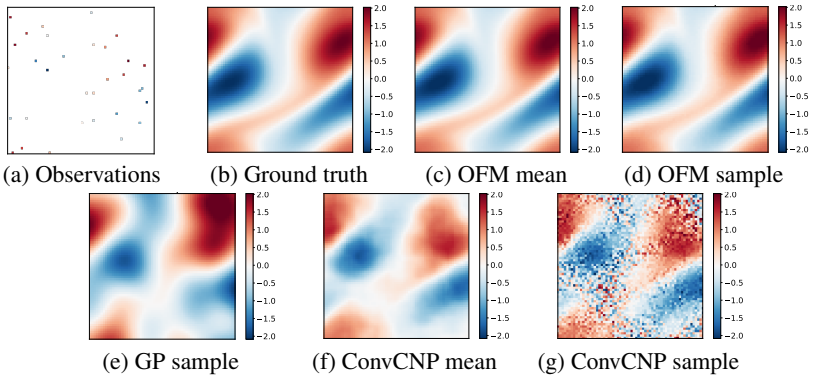
Figure 1: Two-phase strategy for prior learning and posterior sampling. In the prior-learning phase, OFM leverages the marginal optimal-transport path to learn a bijective mapping between a predefined GP and the unknown stochastic process that generates the training data. In the posterior-sampling phase, the learned prior is frozen; given noisy, partial observations, the exact posterior is obtained via Bayes' theorem. SGLD, aided by the Hutchinson trace estimator, then enables efficient and robust sampling.

Lastly, our contributions are summarized as follows:

- **First** work to extend *flow matching / stochastic interpolants / rectified flow* [11–13] to stochastic processes via operator learning. The formulation enables likelihood estimation for function values at any collection of points for the target stochastic process. We also contribute to the development of marginal (dynamic) optimal transport in infinite-dimensional flow matching through optimal coupling and dynamic Kantorovich formulations.
- **First** integration of flow matching with functional regression yields a unified framework for prior and posterior sampling that is applicable to both generation and regression tasks.
- A practical generalization of GP regression that provides the **exact** prior and posterior density over an unknown stochastic process (whether GP or non-GP). In contrast, previous methods such as deep GPs and conditional models work with approximate posteriors. Our method achieves **SOTA** performance in all challenging functional regression tasks.
- This work provides a unified perspective bridging several important fields, which opens new research directions for problems in science and engineering. Additionally, we present extensive ablation and scaling studies that demonstrate the effectiveness of each component in our framework.

2 Related Work

Neural operators. Neural operators constitute a paradigm in machine learning for learning maps between function spaces, a generalization of conventional neural networks that map between finite dimensional spaces [14, 15]. Among neural operator architectures, Fourier neural operators (FNO) [14] enable convolution in the spectral domain and are effective for operator learning [16–21]. In this work, we use this as our neural operator architecture.



(a) Observations (b) Ground truth (c) OFM mean (d) OFM sample

(e) GP sample (f) ConvCNP mean (g) ConvCNP sample

Figure 2: Operator Flow Matching (OFM) regression on Navier-Stokes functional data with resolution 64×64. (a) 32 random observations (only 0.7%). (b) Ground truth sample. (c) Predicted mean from OFM. (d) One posterior sample from OFM. (e) One posterior sample from the best fitted GP. (f) Predicted mean from ConvCNP. (g) One posterior sample from ConvCNP.

Direct function samples. There is a body of work on generative models dedicated to learning distributions over functions, such that direct sampling on the function space is possible. For example, generative adversarial neural operators (GANO) generalize generative adversarial nets on finite dimensional spaces to function spaces [22, 23], yielding a neural operator generative model that maps GP to data functions [3]. Other works in this area have followed the success of diffusion models [24, 25] in finite dimensional spaces, e.g., denoising diffusion operators generalize diffusion models to function spaces by using GP as a means to add noise and use neural operators to learn the score operator on function-valued data [26–28]. Moreover, the same principle has been deployed to generalize flow matching [11] to functional spaces [29], an approach closely related to our work. However, these works on learning generative models on function spaces do not support UFR the way GP-regression does because they (i) focus solely on generating function samples, (ii) do not clarify how to model a stochastic process on point value sequential generation, and (iii) do not provide point evaluation of probability density. In contrast, OFM enables exact density calculation and conditional posterior sampling.

Stochastic processes. Earlier works on SPL have focused on hand-tuned methods in the style of GP-regression. In these cases, an expert tunes the GP parameters given a set of experimental samples. More advanced methods rely on deep GPs, in which a network of GPs is stacked on top of each other. The parameters of deep GPs are commonly optimized by minimizing the variational free energy, which serves as a bound on the negative log marginal likelihood. [30, 31]. Deep GPs have limitations in terms of learnability, expressivity, and computational complexity. Warped GPs [32] and transforming GP [33] methods use historical data to learn a pointwise transformation of GP values and achieve on par performance compared to deep GP type methods. The pointwise nature of such approaches limits their generality.

Another line of work proposes learning the conditional distribution of point values, inspired by variational inference and designed for sampling from function spaces; we refer to this line of models as conditional models[1] [7]. This method trains a model to map any collection of points and their values to a vector, used as an input to a decoder that maps any collection of points to their values. The architectures used in these models (including the decoder) are not mathematically consistent as the number of points grows, limiting the approach to finite dimensions. In particular, Convolutional Conditional Neural Processes (ConvCNPs) [6] use convolutional architectures to capture local, translation-invariant structure, but they assume stationarity and struggle with long-range dependencies or highly non-uniform data. The diffusion based variants [34] also use uncorrelated Gaussian noise, and the results do not exist in function spaces [22, 26]. Furthermore, methods based on conditional models are unable to provide density estimation for collections of points, as needed for UFR and SPL in general.

Separately, OpFlow introduced invertible neural operators that are trained to map any collection of points sampled from a GP to a new collection of points in the data space [1], using the maximum likelihood principle. This method is mathematically consistent as the resolution grows, captures

[1]Neural process (NP) is a prominent example of conditional models

the likelihood of any collection of points, and allows for UFR using SGLD. However, similar to normalizing flow [35] methods in finite dimensional domains, the use of invertible deep learning models makes their training a challenge, particularly with regard to expressiveness, as also described in the original OPFLOW work. Finally, we strongly encourage readers to consult Appendix A and Q for an in-depth comparison of stochastic process learning and other generative frameworks.

3 Operator Flow Matching

Here, we introduce the problem setting and notations used for OFM in function space. We recommend that readers consult Appendix B–E for a foundational overview of SPL, UFR and related background. Subsequently, we present the framework of OFM, which extends marginal optimal transport flow matching [2] to infinite-dimensional spaces. We further demonstrate the generalization of flow matching to stochastic processes as it is induced from OFM on function spaces. Finally, we illustrate how to evaluate exact and tractable likelihoods for any point evaluation of functions using OFM, making it applicable in the UFR setting.

For a real separable Hilbert space $(\mathcal{H}, \langle\cdot,\cdot\rangle, \|\cdot\|)$, equipped with the Borel $\sigma-$ algebra of measurable sets denoted by $\mathcal{B}(\mathcal{H})$, we introduce two measures on $\mathcal{B}(\mathcal{H})$, ν_0 as the reference measure and ν_1 as the data measure. Consider a function h_0 sampled from ν_0, such that $h_0 \sim \nu_0$. For a smooth time-varying infinite dimensional vector field $\mathcal{G}_t : \mathcal{H} \to \mathcal{H}$, we define an ordinary differential equation (ODE)

$$\frac{\partial \Phi_t(h_0)}{\partial t} = \mathcal{G}_t(\Phi_t(h_0)) \tag{1}$$

with initial condition $\Phi_0(h_0) = h_0$, where $h = h_t = \Phi_t(h_0)$ represents a function h_0 transported along a vector field from time 0 to time t. The diffeomorphism Φ_t induces a pushforward measure $\mu_t := [\Phi_t]_\sharp(\mu_0)$, with $\mu_0 = \nu_0$, and we refer to μ_t as the path of probability measure. The goal is to construct a path of probability measure such that at $t = 1$, $\mu_1 \approx \nu_1$. The dynamic relationship between the time varying measure μ_t and vector field $\mathcal{G}_t$ can be characterized by the continuity equation:

$$\frac{\partial \mu_t}{\partial t} = -\nabla \cdot (\mu_t \mathcal{G}_t) \tag{2}$$

In practice, we use Eq. 2 in its weak form [29, 36] to check whether a given vector field $\mathcal{G}_t$ generates the target μ_t:

$$\int_0^1 \int_{\mathcal{H}} \frac{\partial \varphi(h,t)}{\partial t} + \langle \mathcal{G}_t(h), \nabla_h \varphi(h,t)\rangle d\mu_t(h) dt = 0 \tag{3}$$

Where $\varphi \in \text{Cyl}(\mathcal{H} \times [0,1])$, and $\text{Cyl}(\mathcal{H} \times [0,1])$ is the space of smooth cylindrical test functions. Suppose that the time-varying vector field $\mathcal{G}_t$ and the induced μ_t satisfying Eq. 3 are known. We parameterize $\mathcal{G}_t$ with a neural operator $\mathcal{G}_\theta : [0,1] \times \mathcal{H} \to \mathcal{H}$ and regress $\mathcal{G}_\theta$ to target $\mathcal{G}_t$ through flow matching objective.

$$\mathcal{L}^\dagger_{\text{FM}} = \mathbb{E}_{t\sim\mathcal{U}[0,1], h\sim\mu_t} \|\mathcal{G}_\theta(t,h) - \mathcal{G}_t(h)\|^2 \tag{4}$$

However, similar to its finite-dimensional counterpart, $\mathcal{G}_t$ is typically unknown. Moreover, there are infinitely many paths of probability measures that satisfy Eq. 3 and ensure $\mu_1 \approx \nu_1$. Therefore, it is necessary to specify a path of probability measures to effectively guide the learning of $\mathcal{G}_\theta$.

By constructing the appropriate Gaussian and conditional probability measures and leveraging optimal coupling together with the dynamic Kantorovich formulation [37], we propose marginal (dynamic) optimal-transport flow matching in function space; detailed theory development and proofs are provided in Appendix F and G. In the next subsection, we show how to generalize flow matching to stochastic processes, which is induced by the function space derivation.

3.1 Generalizing Flow Matching to Stochastic Processes

Stochastic processes are inherently infinite-dimensional and define distributions over any collection of points (Brémaud [38], Chapter 5.1). We generalize the above marginal optimal transport flow matching on function spaces to stochastic processes by defining the transport map on any collection of points. We then show that, as the collection of points covers the space in the limit, this generalization recovers infinite-dimensional flow matching implemented with neural operators.

For any n and points $\{x_1, x_2, \ldots, x_n\}$, consider an ODE system in which a vector of random variables $u_0 \in \mathbb{R}^n$ is gradually transformed into $u_1 \in \mathbb{R}^n$, for which, the ith entry is equal to $u(x_i)$, via a smooth, time-varying vector field, denoted by $\mathcal{G}_t$ with abuse of notation.

$$u_t := \Phi_t(u_0) = u_0 + \int_0^t \mathcal{G}_s(u_s)ds \tag{5}$$

Here, the neural operator is applied to a collection of point evaluations. Given the set of points and the density of $p_0 := p_0\left(\{u_0(x_1), u_0(x_2), \ldots, u_0(x_n)\}\right) = \mathcal{N}\left(\mathbf{0}, K\left(\{x_1, x_2, \ldots, x_n\}\right)\right)$, where K is a $n \times n$ covariance matrix with entries described by kernel function $k(x_i, x_j)$ and $u_0 \sim p_0$, the time-varying density p_t induced by the diffeomorphism Φ_t or $\mathcal{G}_t$ can be computed, extending the transport equation Eq. 2 to collections of points,

$$\frac{\partial p_t(u_t)}{\partial t} = -(\nabla \cdot (\mathcal{G}_t p_t))(u_t) \tag{6}$$

Eq. 6 shows that constructing p_t is equivalent to constructing $\mathcal{G}_t$ for finite entries for which the analysis carries to finite collections of random variables. In the following, we refer to p_t as the marginal probability path induced by $\mathcal{G}_t$ for the given collection of points. From Eq. 6, the log density can be computed through integration,

$$\log p_t(u_t) = \log p_0(u_0) - \int_0^t (\nabla \cdot \mathcal{G}_s)(u_s)ds \tag{7}$$

In this formulation, we are seeking a specific vector field that transports density q_0 to target density q_1 for any n and any collection of points $\{x_1, x_2, \ldots, x_n\}$ with boundary conditions $p_0 = q_0, p_1 = q_1$. Extending optimal transport flow matching to stochastic processes, we parameterize the vector field $\mathcal{G}_t$ with a neural operator $\mathcal{G}_\theta$, which is optimized through the flow matching objective for SPL,

$$\mathcal{L}_{\text{FM}} := \sup_n \sup_{\{x_1,\ldots,x_n\}} \mathbb{E}_{t\sim\mathcal{U}(0,1),u_t\sim p_t} \|\mathcal{G}_\theta(t, u_t) - \mathcal{G}_t(u_t)\|^2 \tag{8}$$

Note that p_t and u_t depend on the point collocations. In the above equation, the suprema are intractable and we replace them with an expectation as a soft approximation (see Appendix A. Q5 for a detailed discussion of the approximation). Moreover, the true $\mathcal{G}_t$ is usually unknown and to address it, we derive a probability path conditioned on latent variable z of the same alphabet size as the collection. Consequently, the marginal probability path $p_t(u_t)$ is a mixture of conditional probability paths $p_t(u_t|z)$,

$$p_t(u_t) = \int p_t(u_t|z)q(z)dz \tag{9}$$

$$\mathcal{G}_t(u_t) = \mathbb{E}_{q(z)}[\frac{\mathcal{G}_t(u_t|z)p_t(u_t|z)}{p_t(u_t)}]. \tag{10}$$

Given Eq. 10, the conditional flow matching (CFM) objective is defined as,

$$\mathcal{L}_{\text{CFM}} := \mathbb{E}_n \mathbb{E}_{x_1,\ldots,x_n} \mathbb{E}_{t,q(z),p_t(u_t|z)} \|\mathcal{G}_\theta(t, u_t) - \mathcal{G}_t(u_t|z)\|^2. \tag{11}$$

Eqs. 11 and 10 have an identical gradient for θ, i.e. $\nabla_\theta \mathcal{L}_{\text{FM}}(\theta) = \nabla_\theta \mathcal{L}_{\text{CFM}}(\theta)$. Inspired by the finite dimensional developments [2], the variable z is chosen as a couple (u_0, u_1) from the coupling $\pi(u_0, u_1) = q(z)$, which is achieved by minimizing the dynamic 2-Wasserstein distance,

$$W_{\text{dyn}}(q_0, q_1)_2^2 = \inf_{p_t, \mathcal{G}_t} \int_{\mathbb{R}^n} \int_0^1 p_t(u_t)\|\mathcal{G}_t(u_t)\|^2 du_t dt \tag{12}$$

Under mild conditions on $\mathbb{R}^n$, this is equivalent to the static 2-Wasserstein distance,

$$W_{\text{sta}}(q_0, q_1)_2^2 = \inf_{\pi\in\Pi} \int_{\mathbb{R}^n\times\mathbb{R}^n} \|u_1 - u_0\|^2 d\pi(u_0, u_1). \tag{13}$$

Considering the class of Gaussian conditional probability paths $p_t(u_t|z) = \mathcal{N}(u_t|m_t(z), \sigma_t(z)^2 K\left(\{x_1, x_2, \ldots, x_n\}\right))$, with conditional flow $\Phi_t(u_0|z) = \sigma_t u_0 + m_t$. Specially, we choose $m_t = tu_1 + (1-t)u_0$ and $\sigma_t = \sigma$, where $\sigma > 0$ is a small constant. Then we have the following closed-form expression for the corresponding vector field (full derivation provided in Appendix H)

$$\mathcal{G}_t(u_t|z) = u_1 - u_0 \tag{14}$$

With the aforementioned developments, for any collection of points, we transport a Gaussian distribution to a target distribution. The Gaussian distribution is drawn from a GP, with its covariance matrix $K(x_1, \cdots, x_n)$ determined by the kernel function $k(x_i, x_j)$ of the GP. According to Kolmogorov extension theorem (KET) [39], there exists a valid stochastic process whose finite-dimensional marginal is the pushforward distribution under $\mathcal{G}_\theta$. This demonstrates that the generalization of flow matching to infinite-dimensional spaces with neural operators naturally induces the generalization of flow matching to stochastic processes. In the scenario where the limit of points covers the space, these two become equivalent. For a detailed explanation and proof, please refer to Appendix C and D.

Our framework extends seamlessly to alternative probability paths, such as those in *stochastic interpolants and rectified flow* [12, 13], because the generalization to stochastic processes is decoupled from any specific path. However, we focus on the OT path in this work for two reasons: (1) ablation and scaling studies show that it accelerates inference and enables more accurate prior learning compared to independent coupling (Table 7, and Appendix P); (2) theoretically, the OT path (straight line) simplifies the Jacobian evaluation required for likelihood estimation (discussed in the next subsection), yielding greater stability and speed than arbitrary paths.

3.2 Likelihood Estimation and Bayesian Universal Functional Regression

We parameterize $\mathcal{G}_\theta$ with FNO [14] to ensure our model is resolution agnostic, and assume $\mathcal{G}_\theta$ learns a map from ν_0 to ν_1, which serves as the prior. In practice, we deal with discretized evaluations of functions that may have different sampling rate and resolution. For instance, consider a function u sampled from ν_1, observed on a collection of points $u_1 := \{u(x_1), u(x_2), ..., u(x_m)\}$; thus we have a density function $\mathbb{P}(u_1)$ defined on collection of points $\{x_1, x_2, ..., x_m\}$, where $\mathbb{P}(u_1)$ is derived from measure ν_1. This is similar to how a multivariate Gaussian distribution can be derived from a Gaussian measure characterized by a Gaussian process. Therefore, we can rewrite Eq. 7 as:

$$\log \mathbb{P}(u_1) = \log \mathbb{P}(u_0) - \int_0^1 (\nabla \cdot \mathcal{G}_\theta)(u_t) dt \tag{15}$$

where u_0 is drawn from the reference Gaussian measure ν_0, which is also defined on the collection of points $\{x_1, x_2, ..., x_m\}$. Thus, $\mathbb{P}(u_0)$ is a multivariate Gaussian with a tractable density function. Furthermore, with the probability density function $\mathbb{P}(u_1)$, we can evaluate the precise likelihood of any u_1 from $\mathbb{P}(u_1)$ via Eq. 15. However, following a similar argument to Grathwohl et al. [40], the computation of $\nabla \cdot \mathcal{G}_\theta(u)$ incurs a cost of $\mathcal{O}(m^2)$ where m is the cardinality of $\{x_1, x_2, ..., x_m\}$. This quadratic time complexity renders the likelihood calculation prohibitively expensive. To address this issue, we adopt the strategy proposed in Grathwohl et al. [40], using the unbiased Skilling-Hutchinson trace estimator [41, 42] to approximate the divergence term. This technique reduces the computation cost to $\mathcal{O}(m)$, which is the same as the cost of inference, thereby streamlining the evaluation process. The estimator is implemented as follows:

$$\nabla \cdot \mathcal{G}_\theta(u_t) = \mathbb{E}_{p(\varepsilon)}[\varepsilon^T \frac{\partial \mathcal{G}_\theta(u, t)}{\partial u} \varepsilon] \tag{16}$$

In the unbiased trace estimator, the random variable ε is characterized by $\mathbb{E}(\varepsilon) = 0$ and $\text{Cov}(\varepsilon) = I$. This estimator benefits from the optimal transport nature of the map which gives rise to a direct line. The gradient computation in Eq. 16 can be efficiently handled with reverse-mode automatic differentiation, allowing for precise estimation with arbitrary error by averaging over a sufficient number of runs, which can benefit from parallel computing of GPUs.

With the efficient tool established for estimating the likelihood of any discretized function samples, we now turn our attention to UFR, i.e., Bayesian functional regression with a learned prior. Consider a collection of pointwise observations of the underlying unknown function drawn from ν_1, corrupted with Gaussian noise, denoted as $\{\widehat{u}(x_1), \widehat{u}(x_2), \ldots, \widehat{u}(x_n)\}$ or $\{\widehat{u}(x_i)\}_{i=1}^n$. We specifically focus on Gaussian white noise characterized by $\epsilon \sim \mathcal{N}(0, \sigma^2)$, such that $\widehat{u}(x_i) = u(x_i) + \epsilon_i$ for $i \in \{1, \cdots, n\}$ (depending on the nature of the problem, the noise may also be considered as a correlated GP noise). In UFR setting, we are interested in the posterior distribution over new $m \geq n$ points that include the n observation points.

Proposition 3.1. *Given noisy observations $\{\widehat{u}(x_i)\}_{i=1}^n$, the posterior distribution is*

$$\log \mathbb{P}\left(\{u(x_i)\}_{i=1}^m \middle| \{\widehat{u}(x_i)\}_{i=1}^n\right) = -\frac{\sum_{i=1}^n \|\widehat{u}(x_i) - u(x_i)\|^2}{2\sigma^2} + \log \mathbb{P}\left(\{u(x_i)\}_{i=1}^m\right) + C \tag{17}$$

Where the constant $C = -\frac{n}{2}\log(2\pi\sigma^2) - \log \mathbb{P}\left(\{\widehat{u}(x_i)\}_{i=1}^n\right)$.

Proof. This is derived from Bayes' theorem, along with the translation invariance property of the Gaussian distribution, see the full proof in Appendix I □

Given this form of the posterior distribution, we adopt SGLD [4] to efficiently sample from it, and then derive statistical features of interest, e.g. mean, maximum a posteriori, and posterior uncertainty, i.e., variance, from the posterior samples. More specifically, we follow the posterior sampling strategy developed by Shi et al. [1], which, given an invertible framework, suggests SGLD sampling within the input GP space where the Gaussian measure ν_0 is defined and Langevin dynamics is native to, and then mapping to the data function space (where data measure ν_1 is defined). We also present a scaling study in Appendix P analyzing the variance introduced by the Hutchinson trace estimator, demonstrating its robustness and effectiveness when paired with SGLD sampling. Last, Eq 17 offers a unified view of prior and posterior sampling with flow matching models by showing that when no observations are present, the posterior collapses to the prior (up to a constant), making the posterior sampling process identical to that of the prior.

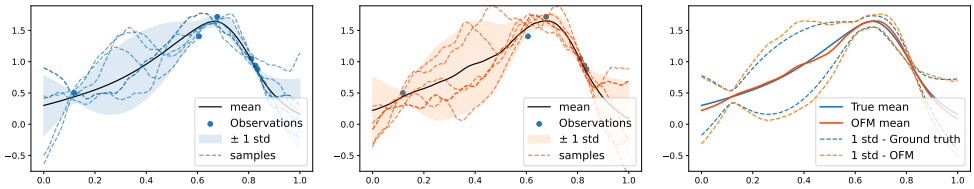
Finally, we provide a plain-language summary of the framework to help the reader better understand the paper (with a detailed discussion available in Appendix Q),

- OFM is an *expressive, flow-based generative model that learns a prior over functions (stochastic process)*: a neural operator parameterizes a continuous probability flow that transports samples from a simple reference Gaussian process to data-like functions, yielding an explicit prior with a tractable density.
- It excels at functional regression by treating functions as *first-class objects* rather than mere pointwise values (unlike NPs), yielding predictions that are consistent across resolutions and arbitrary query sets.
- The learned flow is invertible, enabling change-of-variables for likelihoods and principled Bayesian regression, yielding calibrated uncertainty from few observations.

4 Experiments

In this section, we demonstrate the superior regression performance compared to several baselines across a variety of function datasets, including both Gaussian and highly non-Gaussian Processes. As baselines, we employ standard GP Regression [8], Deep GPs [9, 10], Conditional models [5–7], and OpFlow [1].

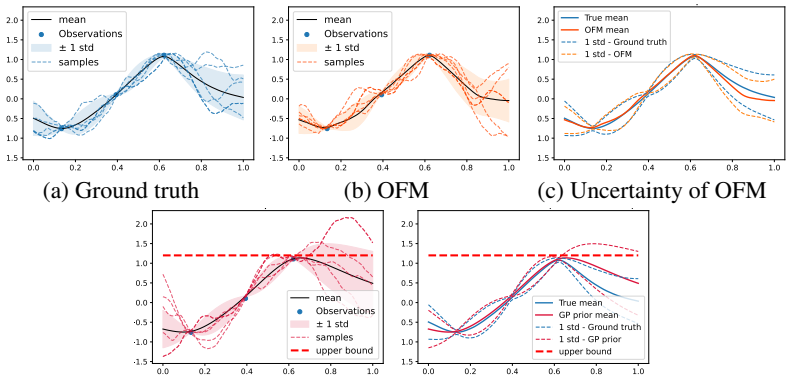
For our function datasets, we analyze: (1) Gaussian and non-Gaussian with known posterior, including 1D GPs, 2D GPs, and 1D Truncated GPs, (TGP). (2) Highly non-GPs, datasets with unknown posterior, such as those derived from Navier-Stokes equations [14], black hole dataset from expensive Monte Carlo simulation, and 2D Signed Distance Functions extracted from MNIST digits (MNIST-SDF) [43]. During regression, we assume that the prior $\mathcal{G}_\theta$ is always successfully trained and remains frozen. Details about the learning process for priors and experimental setup for regression are provided in the Appendix M, O.



(a) Ground truth (b) OFM (c) Uncertainty comparison

Figure 3: OFM regression on GP data. (a) Ground truth GP regression with observed data and predicted samples. (b) OFM regression with observed data and predicted samples. (c) Standard deviation comparison between true GP and OFM predictions.

1D GP data. This experiment replicates the results of classical GP regression, wherein the posterior distributions are precisely known in a closed form. The process involves generating a single new



(a) Ground truth (b) OFM (c) Uncertainty of OFM

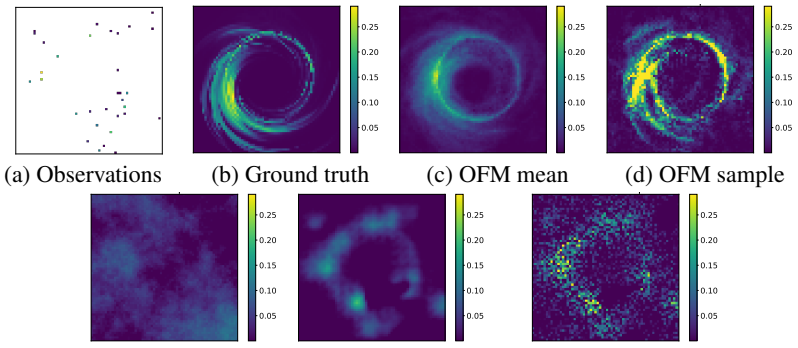
(d) GP prior (e) Uncertainty of GP

Figure 4: OFM regression on TGP data.

realization from the data measure ν_1. We then select observations at $n = 6$ randomly chosen positions, incorporating a predefined noise level. The posterior is inferred across $m = 128$ positions, which includes estimating noise-free values at the observation points. We evaluate our results with two commonly used quantities in the GP literature (1) Standardized Mean Squared Error (SMSE) that normalizes the mean squared error by the variance of the ground truth; and (2) Mean Standardized Log Loss (MSLL), originally introduced by Williams and Rasmussen [8], defined as:

$$-\log p(\{u(x_i)\}_{i=1}^m | \{\widehat{u}(x_i)\}_{i=1}^n) = \frac{1}{2}\log(2\pi\sigma^2) + \frac{(\{u(x_i)\}_{i=1}^m - \{\bar{u}(x_i)\}_{i=1}^m)^2}{2\sigma^2} \tag{18}$$

where $\{\widehat{u}(x_i)\}_{i=1}^n$ represents observations, $\{x_i\}_{i=1}^m, \{u(x_i)\}_{i=1}^m$, indicate the new positions queried, and the test data (true posterior samples). Meanwhile, $\{\bar{u}(x_i)\}_{i=1}^m$ and σ^2 are predicted mean and variances from the model. We average out SMSE and MSLL over a test dataset containing 1000 true GP posterior samples for all models. The performance of each model is detailed in Table 1. From Fig. 3, the regression with OFM matches the analytical solution well and provides realistic posterior samples. We further include GP regression tasks using more complex kernels (Gibbs and Rational Quadratic kernel), as shown in Appendix N, OFM consistently outperforms all comparative methods across all metrics.



(a) Observations (b) Ground truth (c) OFM mean (d) OFM sample

(e) GP sample (f) ConvCNP mean (g) ConvCNP sample

Figure 5: OFM regression on black hole data with resolution 64×64. (a) 32 random observations. (b) Ground truth sample. (c) Predicted mean from OFM. (d) One posterior sample from OFM. (e) One posterior sample from best fitted GP. (f) Predicted mean from ConvCNP. (g) One posterior sample from ConvCNP.

Truncated GP data. In this experiment, we analyze the regression performance of OFM for tractable non-GP. Specifically, we work on truncated GP [1, 44], which constrains the function amplitude within a specified range. This is achieved by applying a sampling-rejection strategy on samples from the GP prior. We set the bounds of our TGP to $[-1.2, 1.2]$ and perform regression using observations only at three points, while estimating the posterior across $m = 128$ points. Subsequently, we sample 1000 true TGP posteriors from the GP prior to calculate the mean and standard deviation. Traditional metrics like MSLL and SMSE, which assume a Gaussian posterior, are not suitable for TGP. We

Table 1: Comparison of OFM with baseline models: GP regression; OpFlow [1]; Conditional models: NP ([7]); Attentive NP ([5], ANP); Convolutional Conditional NP ([6], ConvCNP); Deep variational GP ([9], DGP); Deep Sigma Point Process ([10], DSSP); Metrics SMSE and MSLL used for 1D and 2D GP example. Mean squared error for the predicted mean (μ) and standard deviation (σ) are used for TGP example. Performance of GP regression for 1D and 2D GP are removed (marked with '$-$'), which are taken as the ground truth. Best performance in bold.

Dataset $\rightarrow$	1D GP		2D GP		1D TGP	
Algorithm $\downarrow$ Metric $\rightarrow$	SMSE	MSLL	SMSE	MSLL	μ	σ
GP prior	-	-	-	-	$6.4 \cdot 10^{-2}$	$1.6 \cdot 10^{-2}$
NP	$6.1 \cdot 10^{-1}$	$4.5 \cdot 10^{0}$	$1.7 \cdot 10^{-1}$	$2.1 \cdot 10^{0}$	$1.0 \cdot 10^{-1}$	$1.9 \cdot 10^{-2}$
ANP	$5.1 \cdot 10^{-1}$	$9.8 \cdot 10^{-1}$	$1.6 \cdot 10^{-1}$	$1.1 \cdot 10^{0}$	$1.4 \cdot 10^{-1}$	$1.7 \cdot 10^{-2}$
ConvCNP	$5.6 \cdot 10^{-1}$	$2.7 \cdot 10^{-1}$	$1.7 \cdot 10^{-1}$	$4.5 \cdot 10^{-1}$	$1.6 \cdot 10^{-2}$	$2.1 \cdot 10^{-3}$
DGP	$4.1 \cdot 10^{-1}$	$6.8 \cdot 10^{-2}$	$1.8 \cdot 10^{0}$	$4.2 \cdot 10^{0}$	$4.9 \cdot 10^{-1}$	$1.4 \cdot 10^{-2}$
DSPP	$4.7 \cdot 10^{-1}$	$6.5 \cdot 10^{0}$	$1.9 \cdot 10^{-1}$	$6.6 \cdot 10^{0}$	$1.1 \cdot 10^{-2}$	$1.3 \cdot 10^{-2}$
OpFlow	$5.0 \cdot 10^{-1}$	$2.0 \cdot 10^{-1}$	$1.4 \cdot 10^{-1}$	$\mathbf{1.1 \cdot 10^{-1}}$	$1.3 \cdot 10^{-2}$	$3.9 \cdot 10^{-3}$
OFM(Ours)	$\mathbf{4.1 \cdot 10^{-1}}$	$\mathbf{5.5 \cdot 10^{-2}}$	$\mathbf{1.3 \cdot 10^{-1}}$	$1.6 \cdot 10^{-1}$	$\mathbf{5.2 \cdot 10^{-3}}$	$\mathbf{9.5 \cdot 10^{-4}}$

evaluate the performance using the mean squared error for both the predicted mean and standard deviation. The results are reported in Table. 1, and illustrated in Fig. 4. OFM accurately learns the specified bounds and provides accurate estimations of mean and standard deviation, along with realistic posterior samples. In contrast, directly applying GP regression exceeds the bounds and yields unrealistic posterior samples.

2D GP data. Similar to the 1D GP example, we extend our regression analysis to 2D GP data. As shown in Fig. 6 and detailed in Table 1, OFM provide accurate posterior estimation. The relative error shown in Fig. 6 is the absolute error normalized by the maximum absolute value of the mean prediction derived from the ground truth GP regression.

Navier-Stokes, Black hole and MNIST-SDF datasets. We collected a 2D Navier-Stokes dataset and applied OFM for the regression. Unlike the GP experiments, where MSLL and SMSE score serve as standard benchmarks, evaluating the performance of models on general non-GPs presents a significant challenge due to the difficulty of determining the true posterior and lack of benchmarks. Moreover, the evidence term in the posterior (Eq. 41) is intractable, so the likelihood cannot serve as a meaningful evaluation metric in our setting. A detailed discussion is provided in Appendix A.

We present the predicted mean and a posterior sample in Fig 2 for visual comparison with the ground truth. The predicted mean and the posterior sample are closely aligned with the ground truth. In contrast, traditional GP regression and NP models failed to accurately capture the dynamics of the Navier-Stokes data. In Fig. 5, we conduct a similar analysis using a simulated black hole dataset. Here, OFM provides a much more realistic mean and posterior sample that capture the density and swirling patterns of the black hole. Once again, GP regression and NP fail to capture these key statistics. We observe similar outcomes when applying OFM to the MNIST-SDF example (Fig 8), where OFM correctly recognizes the number "7" while GP regression does not.

5 Conclusion

In this paper, we proposed Operator Flow Matching (OFM) for stochastic process learning, which generalizes flow matching models to infinite-dimensional space and stochastic process with optimal transport path. OFM efficiently computes the probability density for any finite collection of points and supports mathematically tractable functional regression. We extensively tested OFM across a diverse range of datasets, including those with closed-form GP and non-GP data, as well as highly non-GP such as Navier-Stokes and black hole data. In comparative evaluations, OFM consistently outperformed all baseline models, establishing new standards in stochastic process learning and regression.

Despite SOTA accuracy, our method is presently limited to low-dimensional domains and demands larger datasets and more compute than GP-based baselines. see Appendix A, O and Q for details. Python code available at `https://github.com/yzshi5/SPL_OFM`

Acknowledgments

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Science Foundations for Energy Earthshot under Award Number DE-SC0024705. ZER is supported by a fellowship from the David and Lucile Packard Foundation. We also thank Charles Gammie, Ben Prather, Abhishek Joshi, Vedant Dhruv, and Chi-kwan Chan for providing the black hole simulations.

References

[1] Yaozhong Shi, Angela F. Gao, Zachary E. Ross, and Kamyar Azizzadenesheli. Universal Functional Regression with Neural Operator Flows, November 2024. URL http://arxiv.org/abs/2404.02986. arXiv:2404.02986 [cs] version: 3.

[2] Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport, March 2024. URL http://arxiv.org/abs/2302.00482. arXiv:2302.00482 [cs].

[3] Kamyar Azizzadenesheli, Nikola Kovachki, Zongyi Li, Miguel Liu-Schiaffini, Jean Kossaifi, and Anima Anandkumar. Neural operators for accelerating scientific simulations and design. *Nature Reviews Physics*, 6(5):320–328, May 2024. ISSN 2522-5820. doi: 10.1038/s42254-024-00712-5. URL https://www.nature.com/articles/s42254-024-00712-5. Publisher: Nature Publishing Group.

[4] Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 681–688, Madison, WI, USA, June 2011. Omnipress. ISBN 978-1-4503-0619-5.

[5] Hyunjik Kim, Andriy Mnih, Jonathan Schwarz, Marta Garnelo, Ali Eslami, Dan Rosenbaum, Oriol Vinyals, and Yee Whye Teh. Attentive Neural Processes, July 2019. URL http://arxiv.org/abs/1901.05761. arXiv:1901.05761 [cs, stat].

[6] Jonathan Gordon, Wessel P. Bruinsma, Andrew Y. K. Foong, James Requeima, Yann Dubois, and Richard E. Turner. Convolutional Conditional Neural Processes, June 2020. URL http://arxiv.org/abs/1910.13556. arXiv:1910.13556 [stat].

[7] Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J. Rezende, S. M. Ali Eslami, and Yee Whye Teh. Neural Processes, July 2018. URL http://arxiv.org/abs/1807.01622. arXiv:1807.01622 [cs, stat].

[8] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006. Issue: 3.

[9] Hugh Salimbeni and Marc Deisenroth. Doubly stochastic variational inference for deep Gaussian processes. *Advances in neural information processing systems*, 30, 2017.

[10] Martin Jankowiak, Geoff Pleiss, and Jacob R. Gardner. Deep Sigma Point Processes, December 2020. URL http://arxiv.org/abs/2002.09112. arXiv:2002.09112 [cs, stat].

[11] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling, February 2023. URL http://arxiv.org/abs/2210.02747. arXiv:2210.02747 [cs, stat].

[12] Michael S. Albergo and Eric Vanden-Eijnden. Building Normalizing Flows with Stochastic Interpolants, March 2023. URL http://arxiv.org/abs/2209.15571. arXiv:2209.15571 [cs, stat].

[13] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow, September 2022. URL http://arxiv.org/abs/2209.03003. arXiv:2209.03003 [cs].

[14] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier Neural Operator for Parametric Partial Differential Equations, May 2021. URL http://arxiv.org/abs/2010.08895. arXiv:2010.08895 [cs, math].

[15] Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural Operator: Learning Maps Between Function Spaces With Applications to PDEs. *Journal of Machine Learning Research*, 24(89):1–97, 2023. ISSN 1533-7928. URL http://jmlr.org/papers/v24/21-1524.html.

[16] Yan Yang, Angela F. Gao, Jorge C. Castellanos, Zachary E. Ross, Kamyar Azizzadenesheli, and Robert W. Clayton. Seismic wave propagation and inversion with Neural Operators, October 2021. URL http://arxiv.org/abs/2108.05421. arXiv:2108.05421.

[17] Jaideep Pathak, Shashank Subramanian, Peter Harrington, Sanjeev Raja, Ashesh Chattopadhyay, Morteza Mardani, Thorsten Kurth, David Hall, Zongyi Li, Kamyar Azizzadenesheli, Pedram Hassanzadeh, Karthik Kashinath, and Animashree Anandkumar. FourCastNet: A Global Data-driven High-resolution Weather Model using Adaptive Fourier Neural Operators, February 2022. URL http://arxiv.org/abs/2202.11214. arXiv:2202.11214.

[18] Gege Wen, Zongyi Li, Qirui Long, Kamyar Azizzadenesheli, Anima Anandkumar, and Sally M. Benson. Real-time high-resolution CO2 geological storage prediction using nested Fourier neural operators. *Energy & Environmental Science*, 16(4):1732–1741, April 2023. ISSN 1754-5706. doi: 10.1039/D2EE04204E. URL https://pubs.rsc.org/en/content/articlelanding/2023/ee/d2ee04204e. Publisher: The Royal Society of Chemistry.

[19] Yan Yang, Angela F. Gao, Kamyar Azizzadenesheli, Robert W. Clayton, and Zachary E. Ross. Rapid Seismic Waveform Modeling and Inversion with Neural Operators, April 2023. URL http://arxiv.org/abs/2209.11955. arXiv:2209.11955.

[20] Hongyu Sun, Zachary E. Ross, Weiqiang Zhu, and Kamyar Azizzadenesheli. Phase Neural Operator for Multi-Station Picking of Seismic Arrivals. *Geophysical Research Letters*, 50(24):e2023GL106434, 2023. ISSN 1944-8007. doi: 10.1029/2023GL106434. URL https://onlinelibrary.wiley.com/doi/abs/10.1029/2023GL106434. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1029/2023GL106434.

[21] Zongyi Li, Nikola Borislavov Kovachki, Chris Choy, Boyi Li, Jean Kossaifi, Shourya Prakash Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, and Anima Anandkumar. Geometry-Informed Neural Operator for Large-Scale 3D PDEs, September 2023. URL http://arxiv.org/abs/2309.00583. arXiv:2309.00583.

[22] Md Ashiqur Rahman, Manuel A. Florez, Anima Anandkumar, Zachary E. Ross, and Kamyar Azizzadenesheli. Generative Adversarial Neural Operators, October 2022. URL http://arxiv.org/abs/2205.03017. arXiv:2205.03017 [cs, math].

[23] Yaozhong Shi, Grigorios Lavrentiadis, Domniki Asimaki, Zachary E. Ross, and Kamyar Azizzadenesheli. Broadband Ground-Motion Synthesis via Generative Adversarial Neural Operators: Development and Validation. *Bulletin of the Seismological Society of America*, 114(4):2151–2171, August 2024. ISSN 0037-1106, 1943-3573. doi: 10.1785/0120230207. URL https://pubs.geoscienceworld.org/bssa/article/114/4/2151/636448/Broadband-Ground-Motion-Synthesis-via-Generative.

[24] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-Based Generative Modeling through Stochastic Differential Equations, February 2021. URL http://arxiv.org/abs/2011.13456. arXiv:2011.13456 [cs, stat].

[25] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models, December 2020. URL http://arxiv.org/abs/2006.11239. arXiv:2006.11239 [cs, stat].

[26] Jae Hyun Lim, Nikola B. Kovachki, Ricardo Baptista, Christopher Beckham, Kamyar Azizzadenesheli, Jean Kossaifi, Vikram Voleti, Jiaming Song, Karsten Kreis, Jan Kautz, Christopher Pal, Arash Vahdat, and Anima Anandkumar. Score-based Diffusion Models in Function Space, November 2023. URL http://arxiv.org/abs/2302.07400. arXiv:2302.07400 [cs, math, stat].

[27] Jakiw Pidstrigach, Youssef Marzouk, Sebastian Reich, and Sven Wang. Infinite-Dimensional Diffusion Models, October 2023. URL http://arxiv.org/abs/2302.10130. arXiv:2302.10130 [cs, math, stat].

[28] Gavin Kerrigan, Justin Ley, and Padhraic Smyth. Diffusion Generative Models in Infinite Dimensions, February 2023. URL http://arxiv.org/abs/2212.00886. arXiv:2212.00886 [cs, stat].

[29] Gavin Kerrigan, Giosue Migliorini, and Padhraic Smyth. Functional Flow Matching, December 2023. URL http://arxiv.org/abs/2305.17209. arXiv:2305.17209 [cs, stat].

[30] Andreas Damianou and Neil D Lawrence. Deep gaussian processes. pages 207–215. PMLR, 2013.

[31] Haitao Liu, Yew-Soon Ong, Xiaobo Shen, and Jianfei Cai. When Gaussian process meets big data: A review of scalable GPs. *IEEE transactions on neural networks and learning systems*, 31 (11):4405–4423, 2020. ISSN 2162-237X. Publisher: IEEE.

[32] Peng Kou, Feng Gao, and Xiaohong Guan. Sparse online warped Gaussian process for wind power probabilistic forecasting. *Applied energy*, 108:410–428, 2013. ISSN 0306-2619. Publisher: Elsevier.

[33] Juan Maroñas, Oliver Hamelijnck, Jeremias Knoblauch, and Theodoros Damoulas. Transforming Gaussian processes with normalizing flows. pages 1081–1089. PMLR, 2021. ISBN 2640-3498.

[34] Vincent Dutordoir, Alan Saul, Zoubin Ghahramani, and Fergus Simpson. Neural Diffusion Processes, June 2023. URL http://arxiv.org/abs/2206.03992. arXiv:2206.03992 [cs, stat].

[35] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021. ISSN 1533-7928.

[36] Luigi Ambrosio, Nicola Gigli, and Giuseppe Savaré. *Gradient flows: in metric spaces and in the space of probability measures*. Springer Science & Business Media, 2008. ISBN 3-7643-8722-X.

[37] Lénaïc Chizat, Gabriel Peyré, Bernhard Schmitzer, and François-Xavier Vialard. Unbalanced optimal transport: Dynamic and Kantorovich formulations. *Journal of Functional Analysis*, 274(11):3090–3123, June 2018. ISSN 0022-1236. doi: 10.1016/j.jfa.2018.03.008. URL https://www.sciencedirect.com/science/article/pii/S0022123618301058.

[38] Pierre Brémaud. *Probability Theory and Stochastic Processes*. Universitext. Springer International Publishing, Cham, 2020. ISBN 978-3-030-40182-5 978-3-030-40183-2. doi: 10.1007/978-3-030-40183-2. URL http://link.springer.com/10.1007/978-3-030-40183-2.

[39] Andreĭ Nikolaevich Kolmogorov and Albert T Bharucha-Reid. *Foundations of the theory of probability: Second English Edition*. Courier Dover Publications, 2018. ISBN 0-486-82159-5.

[40] Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. FFJORD: Free-form Continuous Dynamics for Scalable Reversible Generative Models, October 2018. URL http://arxiv.org/abs/1810.01367. arXiv:1810.01367 [cs, stat].

[41] Michael F Hutchinson. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076, 1989. ISSN 0361-0918. Publisher: Taylor & Francis.

[42] John Skilling. The eigenvalues of mega-dimensional matrices. *Maximum Entropy and Bayesian Methods: Cambridge, England, 1988*, pages 455–466, 1989. ISSN 9048140447. Publisher: Springer.

[43] Vincent Sitzmann, Eric Chan, Richard Tucker, Noah Snavely, and Gordon Wetzstein. Metasdf: Meta-learning signed distance functions. *Advances in Neural Information Processing Systems*, 33:10136–10147, 2020.

[44] Laura P Swiler, Mamikon Gulian, Ari L Frankel, Cosmin Safta, and John D Jakeman. A survey of constrained Gaussian process regression: Approaches and implementation challenges. *Journal of Machine Learning for Modeling and Computing*, 1(2), 2020. ISSN 2689-3967. Publisher: Begel House Inc.

[45] Gavin Kerrigan, Giosue Migliorini, and Padhraic Smyth. Dynamic Conditional Optimal Transport through Simulation-Free Flows, May 2024. URL http://arxiv.org/abs/2404.04240. arXiv:2404.04240.

[46] Yang Jin, Zhicheng Sun, Ningyuan Li, Kun Xu, Kun Xu, Hao Jiang, Nan Zhuang, Quzhe Huang, Yang Song, Yadong Mu, and Zhouchen Lin. Pyramidal Flow Matching for Efficient Video Generative Modeling, March 2025. URL http://arxiv.org/abs/2410.05954. arXiv:2410.05954 [cs].

[47] Qihao Liu, Xi Yin, Alan Yuille, Andrew Brown, and Mannat Singh. Flowing from Words to Pixels: A Noise-Free Framework for Cross-Modality Evolution, March 2025. URL http://arxiv.org/abs/2412.15213. arXiv:2412.15213 [cs].

[48] Cédric Villani. *Optimal Transport*, volume 338 of *Grundlehren der mathematischen Wissenschaften*. Springer, Berlin, Heidelberg, 2009. ISBN 978-3-540-71049-3 978-3-540-71050-9. doi: 10.1007/978-3-540-71050-9. URL http://link.springer.com/10.1007/978-3-540-71050-9.

[49] Vladimir Igorevich Bogachev. *Gaussian measures*. American Mathematical Soc., 1998. ISBN 0-8218-1054-5. Issue: 62.

[50] Leonid Vasilevich Kantorovich and SG Rubinshtein. On a space of totally additive functions. *Vestnik of the St. Petersburg University: Mathematics*, 13(7):52–59, 1958. ISSN 1063-4541. Publisher: Allerton Press, Inc.

[51] Chunyuan Li, Changyou Chen, David Carlson, and Lawrence Carin. Preconditioned Stochastic Gradient Langevin Dynamics for Deep Neural Networks, December 2015. URL http://arxiv.org/abs/1512.07666. arXiv:1512.07666.

[52] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural Ordinary Differential Equations, December 2019. URL http://arxiv.org/abs/1806.07366. arXiv:1806.07366 [cs, stat].

[53] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using Real NVP, February 2017. URL http://arxiv.org/abs/1605.08803. arXiv:1605.08803.

[54] Jean-David Benamou and Yann Brenier. A computational fluid mechanics solution to the Monge-Kantorovich mass transfer problem. *Numerische Mathematik*, 84(3):375–393, 2000. URL https://www.iap.fr/actualites/laune/2022/TransportOptimal/ark%20_67375_VQC-XB4DR0Z3-2.pdf. Publisher: Springer-Verlag Berlin/Heidelberg.

[55] Emilien Dupont, Yee Whye Teh, and Arnaud Doucet. Generative Models as Distributions of Functions, February 2022. URL http://arxiv.org/abs/2102.04776. arXiv:2102.04776.

[56] Yaozhong Shi, Zachary E. Ross, Domniki Asimaki, and Kamyar Azizzadenesheli. Mesh-Informed Neural Operator : A Transformer Generative Approach, June 2025. URL http://arxiv.org/abs/2506.16656. arXiv:2506.16656 [cs].

[57] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps, October 2022. URL http://arxiv.org/abs/2206.00927. arXiv:2206.00927 [cs].

A Potential questions and answers

Our method combines many vital fields, including

- **Operator learning**
- **Flow-based generative models (flow matching)**; **Optimal transport in function space**
- **Bayesian uncertainty**; **Gradient Markov chain Monte Carlo, SGLD**; **Stochastic trace estimation**
- **Gaussian processes**; **Stochastic processes (Kolmogorov extension theorem)**

Because readers will have diverse backgrounds and expertise, we provide some potential questions and answers to enhance the clarity and highlight our contributions.

Q1: Why is it hard to obtain the true prior and posterior of an unknown stochastic process, and what would we gain if we could?

A1: Generalizing GP regression to arbitrary stochastic process regression has remained a long-standing challenge that has kept many researchers busy for decades. The core difficulties are twofold: first, learning a general stochastic-process prior from historical data with an expressive enough model, and second, deriving both the exact posterior distribution and an efficient sampling scheme.

Prior work falls into three main categories—GPs, deep GPs, and conditional models (NP family). Only the classical GP can fully characterize a stochastic process, and then only when the exact GP prior (mean function, covariance kernel, all hyper-parameters, and observation-noise variance) is already known. Even fitting one GP to data generated by another GP with unknown parameters rarely yields the true posterior. Deep GPs and NPs possess stronger representational power compared to GP, but **representational power $\neq$ exactness**: their optimization rely on approximate posterior and **DO NOT** produce a closed-form "true posterior."

If the true prior and posterior were available, we would gain near-complete control over general stochastic process, enabling perfect uncertainty quantification and Bayes-optimal decisions, which has numerous applications in finance market, science and engineering problems

Q2: How does OFM differ from standard finite-dimensional flow matching? What does it mean to generalize flow matching to stochastic processes, and how does extending a generative model to a function space differ from extending it to a full stochastic process?

A2: A standard flow matching model learns a transport map between two distributions defined on a fixed grid (e.g., a pixel lattice). Consequently, it can only generate samples at that specific resolution. In contrast, OFM learns a map between two stochastic processes in function space. This endows it with several properties that standard flow matching lacks:

(i) **Resolution agnosticism**. OFM learns a transport map between two distributions defined on any given collection of points, without regard for the number of points, or their locations in the domain. This enables capabilities like zero-shot generation without retraining.

(ii) **Stochastic process consistency**. OFM respects the metric of the underlying space, ensuring that points close to each other in the input domain have appropriately correlated values in the output distribution. This is part of learning a valid stochastic process, which also satisfies crucial theoretical properties like the Kolmogorov extension theorem (i.e, consistency under marginalization)

(iii) **Convergence to a continuous function**. As the collection of query points becomes denser within the domain, the output of OFM converges to the underlying continuous function.

(iv) **Backwards compatibility**. When evaluated on a fixed set of points, OFM recovers the behavior of a conventional (finite-dimensional) flow matching up to a linear transformation.

Extending a generative model (e.g., flow matching) to an *infinite-dimensional function space* and extending it to a *full stochastic process* are related but distinct goals. The first concerns generating function values at any finite collection of points; Lebesgue measure and Kolmogorov consistency come into play, but the focus remains on finite dimensional marginals. The second addresses the

distribution of the entire function itself, requiring a probability measure over an infinite-dimensional space.

Because of this distinction, lifting a model merely to a *function space* is strictly weaker. For example, GANO [22] extends GANs to function space but not to a stochastic process, as it cannot describe the joint law of function values at arbitrary query sets. In OFM, we bridge these two viewpoints for flow matching with neural operators: by exploiting the discretization-convergence property of neural operators and the invertibility of flow matching, we extend the method to both function spaces and stochastic processes. See Appendix C and D for details.

Q3: These methods may appear similar to image inpainting or restoration with flow- or diffusion-based models; could you elaborate on the differences?

A3: Current inpainting with flow/diffusion models treats the task as an inverse problem in a finite-dimensional setting: one learns to reconstruct missing pixels on a fixed resolution grid, effectively solving a regression problem at a predetermined set of points. By contrast, OFM operates directly on *functions* and is *resolution-agnostic*. Given the same partial observations, our model can predict the entire function at any collection of query points, coarse or fine.

Moreover, OFM delivers principled uncertainty: its posterior quantifies the full distribution of possible completions, whereas finite-dimensional inpainting methods typically rely on an ensemble of visually plausible samples whose variability is not a calibrated measure of uncertainty. Finally, OFM remains effective even when observations are extremely sparse. (e.g 0.7% of total observations) —a regime in which grid-based inpainting approaches struggle.

Q4 : Could you elaborate further on the connections to related work? The approach appears to intersect with several studies, including OpFlow [1], COT-FM [45], OT-CFM [2], and others.

A4: Because our work combines multiple fields, it naturally has connections with many other studies. Readers are referred to the appendix Q for a detailed discussion.

Q5: In Eq 11. What is the argument of replacing the superma in Eq 8 with expecations? When is this valid?

A5: The supremum in Eq. 8 represents a worst-case error over the entire function space, which is computationally intractable to optimize directly. We therefore relax this hard constraint by replacing the supremum with an expectation, as formulated in Eq 11. This is a common empirical consideration.

Instead of minimizing the worst-case error, our objective becomes minimizing the tractable average-case error across the distribution. Minimizing the error on average provides a strong practical incentive for the model to perform well across the entire function space, thereby effectively reducing the worst-case error. Such replacement is always valid under the weaker goal. The validity of this approach as a tractable proxy is further confirmed by our empirical results, which show it successfully guides the model to learn the intended functional mapping (Appendix M).

Q6: Why not use log-likelihood as the evaluation metric for non-GP regression tasks when comparing to NP models? And why not include more recent conditional model baselines, such as NDP [34]

A6: There are several reasons that likelihood as an evaluation metric is not relevant in our case. First, as shown in Eq 41, there is an evidence term, which is a constant, in the posterior distribution in OFM framework. The constant evidence term is intractable but does not contribute to MAP estimation, mean estimation, and posterior sample in general.

Second, even if we can compute the evidence term, we still cannot make the comparison. The graphical model in the NP is such that the conditional model is trained using the MLE principle and the learned likelihood model dependent. The posterior in OFM provides the posterior using the Bayes rule, utilizes a different model, and has a different graphical model. These quantities are not directly relevant to be compared.

Third, computing the true posterior for a general stochastic process is a known challenging problem. Due to the complexity nature of the problem, there doesn't exist a well-recognized metric. The

evaluation of quality of posterior performance requires domain knowledge from experts and varies case by case, which is an exciting research direction.

For baseline models,We already include a broad set of SOTA baselines. On the operator-learning side, the latest OPFLOW [1] (2024) is covered. For deep GPs, we adopt Doubly Stochastic Variational Deep GPs [9] and the Deep Sigma-Point Process [10], both widely recognized and backed by publicly reproducible code.

Within the NP family, ConvCNP [6] remains a standard benchmark, and we include two additional NP variants. Our baselines are limited to models with publicly reproducible implementations. Although we attempted to add the newer NDP [34], we encountered significant reproducibility issues with the authors' code—an obstacle reported by others as well. Instead, we offer a detailed theoretical comparison between OFM and NDP in Appendix Q.

Q7: What do you mean by an "exact posterior" and a practical solution? How can I apply the model to my own tasks, and what do I need to prepare?

A7: For functional regression, posterior error has two sources for any method: **(1) formulation error**—the gap between the model's theoretical posterior and the true one—and **(2) approximation error** from finite data, limited capacity, and optimization. Deep GPs rely on variational inference, optimizing an ELBO and thereby introducing a formulation gap: the posterior is only an *approximation*. NPs make even stronger simplifying assumptions, and doesn't provide the true posterior. All models, including OFM, incur approximation error, but the discussion above concerns only **formulation error**.

A *practical* solution means an expressive backbone that can learn a complex stochastic-process prior and a posterior-sampling routine whose runtime and memory footprint remain acceptable for typical users; see Appendix O for quantitative details.

Using the model is straightforward. OFM offers GP-style regression for non-GP tasks: given noisy observations, it returns the posterior at arbitrary query points. The key difference is that a GP prior is fixed by a hand-tuned kernel, whereas OFM learns the stochastic-process prior directly from data. For details on prior learning and SGLD-based posterior sampling, see Appendices M and L.

B Background: Flow Matching, Gaussian Measures on Function Spaces, and the Cameron–Martin Theorem

In this section, we provide essential background and high-level intuitive explanations of the topics involved in this paper for readers.

Flow Matching. Flow matching is a state-of-the-art generative paradigm that learns a time-dependent velocity field whose ODE transports samples from a simple reference (e.g., Gaussian) to the target data distribution, yielding an unbiased, simulation-free training objective that directly regresses ground-truth velocities along probability paths.

The approach is tightly linked to physics via the continuity equation, and—when cast as a continuous normalizing flow—provides a (deterministically) invertible transformation with tractable likelihoods, while also admitting stochastic variants when desired. In practice it matches diffusion-level quality with faster training and sampling (often few- or even single-step generation) and excellent scalability, which is why it has been adopted in challenging domains such as large-scale video generation, in-context image generation, and protein ensemble generation [46, 47]. We therefore use flow matching as the foundation for prior learning, leveraging its simple training objective, physics-grounded structure, and strong empirical performance.

Gaussian measures. A probability measure μ on a separable Hilbert space $\mathcal{H}$ is *Gaussian* if for any finite collection of vectors $\{h_1, \dots, h_n\} \subset \mathcal{H}$, the random vector $(\langle X, h_1\rangle, \dots, \langle X, h_n\rangle)$ has a multivariate Gaussian distribution, where $X \sim \mu$. The family $\{\langle X, h\rangle : h \in \mathcal{H}\}$ is therefore a *Gaussian process* indexed by $\mathcal{H}$, with mean $h \mapsto \langle m, h\rangle$ for some $m \in \mathcal{H}$. Conversely, if a Gaussian process $\{X(t) : t \in T\}$ has sample paths that almost surely belong to a function space $\mathcal{H}$ (e.g., $C(T)$ or $L^2(T)$), then the law of the random path $t \mapsto X(t)$ is a *Gaussian measure* on $\mathcal{H}$. In short: a Gaussian measure is the path-space law of a GP, and a GP is the collection of linear probes of a Gaussian measure.

Cameron–Martin space and theorem. Let μ be a Gaussian measure on a separable Hilbert space $\mathcal{H}$ with mean $m \in \mathcal{H}$ and covariance operator $C : \mathcal{H} \to \mathcal{H}$ (self-adjoint, positive, trace-class). The *Cameron–Martin space* (the RKHS associated with μ) is

$$\mathcal{H}_\mu \;=\; \overline{\mathrm{Range}(C^{1/2})} \subseteq \mathcal{H},$$

equipped with the inner product $\langle u, v\rangle_{\mathcal{H}_\mu} := \langle C^{-1/2}u,\, C^{-1/2}v\rangle_{\mathcal{H}}$ on $\mathrm{Range}(C^{1/2})$, extended by completion. With this choice, the inclusion $\mathcal{H}_\mu \hookrightarrow \mathcal{H}$ is continuous. For $h \in \mathcal{H}$, write $\mu_h(A) := \mu(A - h)$ for the translate. The *Cameron–Martin theorem* states:

(i) If $h \in \mathcal{H}_\mu$, then μ_h is absolutely continuous with respect to μ (in fact, μ_h and μ are equivalent), with

$$\frac{d\mu_h}{d\mu}(x) = \exp\Big(\big\langle C^{-1/2}h,\, C^{-1/2}(x-m)\big\rangle_{\mathcal{H}} \;-\; \tfrac{1}{2}\,\|h\|^2_{\mathcal{H}_\mu}\Big) \quad \text{for } \mu\text{-a.e. } x,$$

where the pairing is understood in the Paley–Wiener sense (which in finite dimensions reduces to $\langle h,\, C^{-1}(x-m)\rangle$).

(ii) If $h \notin \mathcal{H}_\mu$, then $\mu_h \perp \mu$ (mutually singular).

In finite dimensions this reduces to the classical mean-shift formula for $\mathcal{N}(m, \Sigma)$ with $C = \Sigma$, and $\|h\|^2_{\mathcal{H}_\mu} = \langle h, \Sigma^{-1}h\rangle$. Thus, $\mathcal{H}_\mu$ pinpoints exactly the directions along which a Gaussian measure can be translated while remaining (mutually) absolutely continuous.

C Stochastic process learning

Let $(\Omega, \mathcal{F}, P)$ denote a probability space and let $(\mathbb{R}^d, \mathcal{B}(\mathbb{R}^d))$ denote a measurable space where $\mathcal{B}(\mathbb{R})$ is the Borel space. Following the standard definition of stochastic processes (Brémaud [38], Chapter 5.1), a stochastic process $\mathcal{P}$ on a domain D is a collection of $\mathbb{R}^d$-valued random variables indexed by members of D, i.e.,

$$\{a(x) : x \in D\}$$

jointly following the probability law P. In the special case of Gaussian processes, e.g., Wiener process, following the Gaussian law for P, for any collection points $\{x_1, x_2, \ldots, x_n\}$, the random variables $\{a(x_1), a(x_2), \ldots, a(x_n)\}$ are jointly Gaussian, resulting in a function a to be drawn from a GP. We need to emphasize, $\{a(x_1), a(x_2), \ldots, a(x_n)\}$ is a collection of random variables (random vector) equipped with Lebesgue measure, and represents a discretized observation of one continuous function a. In practice, the joint probability distribution of the collection of the random variables is unknown a priori, and needs to be learned.

In SPL, one way we suggest is to learn an invertible operator $\mathcal{T}$ that maps a base stochastic process $\mathcal{P}$ to another stochastic process $\mathcal{Q}$ that represents the data via discretization convergence theorem (see Appendix D). That is, for any collection of points $\{x_1, x_2, \ldots, x_n\}$, and for any n, the operator $\mathcal{T}$ maps the law on $\{a(x_1), a(x_2), \ldots, a(x_n)\}$ to $\{u(x_1), u(x_2), \ldots, u(x_n)\}$ and vice versa for the inverse $\mathcal{T}^{-1}$, where $u(x)$ is a pointwise evaluation of function data sample, i.e.,

$$\{u(x_1), u(x_2), \ldots, u(x_n)\} = \mathcal{T}\left(\{a(x_1), a(x_2), \ldots, a(x_n)\}\right)$$

Then, the probability of $\{u(x_1), u(x_2), \ldots, u(x_n)\}$, at evaluation points $\{x_1, x_2, \ldots, x_n\}$, for any n and collection of points on D is given by,

$$\mathbb{P}\left(\{u(x_1), u(x_2), \ldots, u(x_n)\}\right) = \mathbf{J}\mathcal{T}\Big|_{\{a(x_1), a(x_2), \ldots, a(x_n)\}} \mathbb{P}\left(\{a(x_1), a(x_2), \ldots, a(x_n)\}\right)$$

where with abuse of notation $\mathbb{P}(u(x))$ denotes the density of $u(x)$ at point x, same for $\mathbb{P}(a(x))$, and similarly, following the notation in Theorem 11.1 of Villani [48], $\mathbf{J}\mathcal{T}\Big|_{\{a(x_1), a(x_2), \ldots, a(x_n)\}}$ is the absolute value of the Jacobian determinant of the map from the random vector $\{a(x_1), a(x_2), \ldots, a(x_n)\}$ at points $\{x_1, x_2, \ldots, x_n\}$ to the random vector $\{u(x_1), u(x_2), \ldots, u(x_n)\}$ via inverse operator $\mathcal{T}^{-1}$. We further show that the pushedforward $\mathcal{Q}$ is indeed a valid stochastic process via Kolmogorov Extension Theorem (KET) [39] with a proof provided in Appendix. D . In SPL, we aim to learn a neural operator $\mathcal{T}_\theta$ such that the resulting $\mathcal{Q}$ matches the data process under the true $\mathcal{T}$.

D Model stochastic process with infinite-dimensional flow matching via Kolmogorov Extension Theorem

In operator learning, neural operators [3, 14, 15] are typically designed to map an input function to an output function. When the input function is provided at a specific discretization (e.g., a set of points with their corresponding values), the model processes this discretized input as a collection of points and their values. Traditionally, in operator learning, this process is seen as an approximation of the operator's application to the underlying continuous function, where the discretization introduces approximation errors. Thus, the input is conceptually still treated as a function.

Moreover, the application of the operator to a collection of points is well-defined, and, by the discretization convergence theorem, as the number of points increases, this operation converges to a well-defined mapping. In this paper, leveraging these properties, we adopt a different perspective as described in the introduction. We extend neural operators to define explicit maps between collections of points. In this framework, the input is not the abstract function itself but rather a collection of points and their associated values. Importantly, this mapping remains well-defined regardless of the number of points in the collection and, by the discretization convergence theorem, converges to a unique mapping as the point collection approaches the underlying continuous function.

Next, we show that given an invertible operator $\mathcal{T}$ and a valid stochastic process $\mathcal{P}$ whose finite dimensional marginal is $\mathbb{P}(\{a(x_1), a(x_2), ..., a(x_n)\}$, there exist a valid stochastic process $\mathcal{Q}$ with finite-dimensional marginal $\{u(x_1), u(x_2), \ldots, u(x_n)\}$.

Once again, as defined in Section C

$$\{u(x_1), u(x_2), \ldots, u(x_n)\} = \mathcal{T}\left(\{a(x_1), a(x_2), \ldots, a(x_n)\}\right)$$

Then, the probability of $\{u(x_1), u(x_2), \ldots, u(x_n)\}$, at evaluation points $\{x_1, x_2, \ldots, x_n\}$, for any n and collection of points on D is given by,

$$\mathbb{P}\left(\{u(x_1), u(x_2), \ldots, u(x_n)\}\right) = \mathbf{J}\mathcal{T}\Big|_{\{a(x_1), a(x_2), \ldots, a(x_n)\}} \mathbb{P}\left(\{a(x_1), a(x_2), \ldots, a(x_n)\}\right) \quad (19)$$

where with abuse of notation $\mathbb{P}(u(x))$ denotes the density of $u(x)$ at point x, same for $\mathbb{P}(a(x))$, and similarly, following the notation in Theorem 11.1 of Villani [48], $\mathbf{J}\mathcal{T}\Big|_{\{a(x_1), a(x_2), \ldots, a(x_n)\}}$ is the absolute value of the Jacobian determinant of the map from the random vector Jacobian of the map from the random vector $\{a(x_1), a(x_2), \ldots, a(x_n)\}$ at points $\{x_1, x_2, \ldots, x_n\}$ to the random vector $\{u(x_1), u(x_2), \ldots, u(x_n)\}$ via inverse operator $\mathcal{T}^{-1}$. We should notice Eq. 19 represents the changes of variables between two random vectors, with Lebesgue measure involved. The connection between the finite-dimensional marginal (equipped with Lebesgue measure) and the probability measure of a stochastic process in infinite-dimensional space is described by Kolmogorov Extension theorem (KET) [39], which assures that if all finite-dimensional distributions (i.e., distributions of function at finite collection of points) are consistent, then a stochastic process exists that matches finite-dimensional distributions.

Formally, according to KET, to establish that a valid stochastic process $\mathcal{Q}$, which has $\mathbb{P}\left(\{u(x_1), u(x_2), \ldots, u(x_n)\}\right)$ as its finite dimensional distributions, it is essential to demonstrate that such a joint distribution satisfies the following two consistency properties:

Permutation invariance. For any permutation π of $\{1, \cdots, n\}$, the joint distribution should remain invariant when elements of $\{x_1, \cdots, x_n\}$ are permuted, such that

$$\mathbb{P}\left(\{u(x_1), u(x_2), \ldots, u(x_n)\}\right) = \mathbb{P}\left(\{u(x_{\pi(1)}), u(x_{\pi(2)}), \ldots, u(x_{\pi(n)})\}\right) \quad (20)$$

Marginal Consistency. This principle specifies that that if a portion of the set is marginalized, the marginal distribution will still align with the distribution defined on the original set, such that for $m \geq n$

$$\mathbb{P}\left(\{u(x_1), u(x_2), \ldots, u(x_n)\}\right) = \int \mathbb{P}\left(\{u(x_1), u(x_2), \ldots, u(x_m)\}\right) du(x_{n+1}) \cdots du(x_m) \quad (21)$$

The permutation invariance property is naturally upheld when utilizing operator, as there is no inherent order among the elements in the set $\{x_1, x_2, \ldots, x_n\}$. Furthermore, the marginal

consistency property is also maintained due to the definition of operator $\mathcal{T}$ (see Eq. 19), which ensures that $\mathbb{P}(\{u(x_1), u(x_2), \ldots, u(x_n)\})$ is closed under marginalization. This is because $\mathbb{P}(\{a(x_1), a(x_2), \ldots, a(x_n)\})$ is closed under marginalization, which fully determines $\mathbb{P}(\{u(x_1), u(x_2), \ldots, u(x_n)\})$ through the Jacobian. While verifying that $\mathcal{Q}$ constitutes a valid induced stochastic process is straightforward given the $\mathcal{T}$, approximating the $\mathcal{T}$ with a neural operator is non-trivial and depends highly on the model used (related to expressiveness). For instance, in Transforming GP [33], the authors employ a marginal normalizing flow, which acts as a point-wise operator to transform values from a GP to another. Consequently, the induced Jacobian is a diagonal matrix. More recently, OpFlow [1] introduces an invertible neural operator by generalizing RealNVP to function space, which induces a triangular Jacobian matrix. In our work, we extend this framework to a more comprehensive case: a diffeomorphism. Here, the induced Jacobian is a full-rank matrix and is not necessarily triangular or diagonal, the determinant of the Jacobian for any collection of points is calculated through Eq 15.

Last, we want to clarify the the connection between the notions of operator $\mathcal{T}$ and operator $\mathcal{G}$ throughout this paper. The operator $\mathcal{T}$ is the Φ_t (a diffeomporhism) defined in Eq. 5, which is the integral of $\mathcal{G}$ over time interval $[0, 1]$. Due to the nature of an ODE system, $\mathcal{T}$ is invertible. However, $\mathcal{G}$ is not necessary invertible, which enables us to parameterize it with a classical neural operator, like FNO [14].

E Universal Functional Regression

UFR is concerned with Bayesian regression on function spaces [1], where it can be used to infer the posterior of an unknown function on a domain D from a collection of pointwise observations. The observations are often corrupted with noise of variance σ^2, denoted as $\{\widehat{u}(x_1), \widehat{u}(x_2), \ldots, \widehat{u}(x_n)\}$ or $\{\widehat{u}(x_i)\}_{i=1}^n$. More specifically, for $m \geq n$ points at which the function is to be inferred,

$$\mathbb{P}\left(\{u(x_1), u(x_2), \ldots, u(x_m)\} \middle| \{\widehat{u}(x_1), \widehat{u}(x_2), \ldots, \widehat{u}(x_n)\}\right)$$

Note that when the prior over the function space is Gaussian, UFR reduces to the celebrated GP regression. Following Bayes rule, and maps between stochastic processes, we obtain the log posterior as follows,

$$\begin{aligned}\log \mathbb{P}\left(\{u(x_i)\}_{i=1}^m \middle| \{\widehat{u}(x_i)\}_{i=1}^n\right) = &-\frac{1}{2}\sum_i^n \frac{\left(\widehat{u}(x_i) - u(x_i)\right)^2}{\sigma^2} - n\log(\sigma) - \frac{n}{2}\log(2\pi) \\ &+ \log \mathbb{P}\left(\{u(x_i)\}_{i=1}^m\right) - \log \mathbb{P}\left(\{\widehat{u}(x_i)\}_{i=1}^n\right)\end{aligned}$$

This equality holds for any collection of points. It is worth noting that the posterior is exact up to constants, i.e., the second, third, and last terms are constant. Therefore, they do not contribute in MAP estimation, mean estimation, and functional regression in general, and there is no need to compute them.

F Marginal (dynamic) optimal-transport flow matching in function space via optimal coupling and dynamic Kantorovich formulation

Consider a joint probability measure $\pi(\nu_0, \nu_1)$ on $\mathcal{H} \times \mathcal{H}$, where the reference measure ν_0, is chosen as a Gaussian measure, whose absolute continuity is well-studied [49]. We characterize ν_0 by a GP with trace-class covariance operator. e.g. $\nu_0 = \mathcal{N}(m_0, C_0)$, where m_0 is the mean, C_0 is the covariance operator. With the joint measure $\pi(\nu_0, \nu_1)$, we sample a function pair $z := (h_0, h_1)$.

Assuming ν_1 has full support on the Cameron-Martin space associated with ν_0 (following the convention of literatures [26, 29]), we construct a conditional probability measure $\mu_t(\cdot|z)$ as a Gaussian measure with trace-class covariance operator and small operator norm to approximate Dirac measures in the sense of weak convergence. Such that, at $t = 0$ and $t = 1$, $\mu_t(\cdot|z)$ is a centered around h_0, h_1, approximating $\delta_{h_0}, \delta_{h_1}$ respectively; Subsequently, we can construct a new marginal probability measure by mixing these approximated Dirac measures:

$$\mu_t(A) = \int \mu_t(A|z) d\pi(z), \ \forall A \in \mathcal{B}(\mathcal{H}) \tag{22}$$

Due to $d\pi(z)$ being always positive, the conditional probability measure (Dirac measure approximated by Gaussian measure) is absolutely continuous with respect to μ_t. Eq. 22 indicates that $\mu_0 = \int \delta_{h_0} d\pi(z) \approx \nu_0$, and $\mu_1 = \int \delta_{h_1} d\pi(z) \approx \nu_1$. This formulation suggests that μ_0, μ_1 represent convolutions of ν_0, ν_1 with Gaussian measures. For a more detailed discussion on convolution with Gaussian measures, we refer the readers to Appendix B.1 of Lim et al. [26].

Please note, dirac measure is not a necessary condition for Eq. 22; the only constraint is the boundary conditions. One viable choice for the conditional measure is a Gaussian measure with a small operator norm, which (in fact) approximates the Dirac measure in the weak convergence sense. Alternative probability path, as described in Lipman et al. [11], Albergo and Vanden-Eijnden [12], Liu et al. [13] are equally valid.

Suppose $\int_0^1 \int_{\mathcal{H}} \int_{\mathcal{H}\times\mathcal{H}} \|\mathcal{G}_t(h|z)\| d\mu_t(h|z) d\pi(z) dt$ is finite to guarantee the vector field is sufficiently regular (Lipschitz continuity), where $\mathcal{G}_t(\cdot|z)$ is the conditional vector field. Under this condition, the vector field that generates μ_t as specified in Eq. 22 and Eq. 3 can be expanded as follows :

$$\mathcal{G}_t(h) = \int_{\mathcal{H}\times\mathcal{H}} \mathcal{G}_t(h|z) \frac{d\mu_t(\cdot|z)}{d\mu_t}(h) d\pi(z) \tag{23}$$

Eq. 23 is an extension of the Theorem 1 as detailed in Kerrigan et al. [29], and we provide the derivation in Appendix G. We note that $\mu_t(\cdot|z)$ is a Gaussian measure and can be expressed as $\mu_t(\cdot|z) = \mathcal{N}(m_t, C_t)$, with mean m_t and trace-class covariance operator C_t. Inspired by Tong et al. [2], we choose m_t and C_t to have the following forms:

$$m_t = t \cdot h_1 + (1-t) \cdot h_0 \tag{24}$$

$$C_t = \sigma_{\text{min}}^2 C_0 \tag{25}$$

where C_0 is the same Gaussian covariance operator defined for ν_0 and σ_{min} is a small constant. Further, similar to finite-dimensional flow matching, we only consider the simplest vector field that applies a canonical transformation for Gaussian measures, such that the flow has the form: $\Phi_t(h_0|z) = m_t + \sigma_{\text{min}} h_0 \approx t \cdot h_1 + (1-t) \cdot h_0$. From Eq. 1, we can get $\mathcal{G}_t(h|z) = h_1 - h_0$, indicating $\mathcal{G}_t(h|z)$ is independent of the time t and the path from h_0 to h_1 is a direct, straight line. Equipped with well-constructed conditional vector field and probability measures, we can train a neural operator $\mathcal{G}_\theta$ with the conditional flow matching loss

$$\mathcal{L}_{\text{CFM}}^\dagger = \mathbb{E}_{t\sim\mathcal{U}[0,1], h\sim\mu_t, z\sim\pi(\nu_0,\nu_1)} \|\mathcal{G}_\theta(t,h) - \mathcal{G}_t(h|z)\|^2 \tag{26}$$

Next, we explore how to approximate the true optimal transport plan from optimal coupling of the joint measure $\pi(\nu_0, \nu_1)$. A common way for measuring the distance between two probability measure is 2-Wasserstein distance, which a special case of static Kantorovich formulation [50]. The static 2-Wasserstein distance is defined as follows

$$W_{\text{sta}}(\nu_0, \nu_1)_2^2 = \inf_{\pi\in\Pi} \int_{\mathcal{H}\times\mathcal{H}} \|h_0 - h_1\|^2 d\pi(h_0, h_1) \tag{27}$$

In the ODE framework, we also care about the dynamic form of the 2-Wasserstein distance to estimate the cost along the transport trajectory, which also is a special case of dynamic Kantorovich formulation [37].

$$W_{\text{dyn}}(\nu_0, \nu_1)_2^2 = \inf_{\mu_t, \mathcal{G}_t} \int_{\mathcal{H}} \int_0^1 \|\mathcal{G}_t(h)\|^2 d\mu_t(h) dt \tag{28}$$

Within the OFM framework, the marginal probability measure is a sum of Dirac measures as described in Eq. 22, and we selected ν_0 as a Gaussian measure and assumed ν_1 has full support on the Cameron-Martin space associated with ν_0. Furthermore, the cost function of 2-Wasserstein distance is squared L^2 norm, which is continuous by nature. According to Theorem 4.3 and Lemma 4.4 of Chizat et al. [37], $W_{\text{sta}} = W_{\text{dyn}}$ for our specifically constructed μ_t and $\mathcal{G}_t$ in the sense of weak convergence. Therefore, to get the dynamic optimal transport plan, we only need to find a joint measure $\pi(\nu_0, \nu_1)$ that achieves the infimum in Eq. 27. In practice, we use a minibatch approximation of optimal coupling between ν_0 and ν_1. The above approach extends the dynamic (marginal) optimal transport framework of [2] to infinite-dimensional function space. The related work of Kerrigan et al. [45] addresses a similar problem, but from a different perspective. For a detailed comparison, please refer to Appendix Q.

G Derivation of Eq. 23

In this part, we show the derivation of Eq. 23, which extends Theorem 1 of Kerrigan et al. [29]. The problem setting is given continuity equation and its weak form:

$$\int_0^1 \int_{\mathcal{H}} \frac{\partial \varphi(h,t)}{\partial t} + \langle \mathcal{G}_t(h), \nabla_h \varphi(h,t) \rangle d\mu_t(h) dt = 0, \quad \forall \varphi \in \text{Cyl}(\mathcal{H} \times [0,1]) \tag{29}$$

we want to derive the following form of the conditional vector field under absolute continuity assumption and other mild conditions, where $z := (h_0, h_1) \in \mathcal{H} \times \mathcal{H}$.

$$\mathcal{G}_t(h) = \int_{\mathcal{H}\times\mathcal{H}} \mathcal{G}_t(h|z) \frac{d\mu_t(\cdot|z)}{d\mu_t}(h) d\pi(z) \tag{30}$$

First, $\int_0^1 \int_{\mathcal{H}} \frac{\partial \varphi(h,t)}{\partial t} d\mu_t(h) dt = \int_0^1 \int_{\mathcal{H}} \int_z \frac{\partial \varphi(h,t)}{\partial t} d\mu_t(h|z) d\pi(z) dt$. With continuity equation in strong form and the fact that $\mathcal{G}_t(h|z)$ induces $\mu_t(h|z)$ we have:

$$\int_0^1 \int_{\mathcal{H}} \int_z \frac{\partial \varphi(h,t)}{\partial t} d\mu_t(h|z) d\pi(z) dt = \int_0^1 \int_{\mathcal{H}} \int_z -\nabla \cdot (\varphi(h,t) \mathcal{G}_t(h|z)) d\mu_t(h|z) d\pi(z) dt$$

By the divergence-form identity:

$$\nabla \cdot (\varphi(h,t)\mathcal{G}_t(h|z)) = \langle \mathcal{G}_t(h), \nabla_h \varphi(h,t) \rangle + \varphi(h,t) \nabla \cdot \mathcal{G}_t(h|z)$$

Since we choose the smooth test function $\varphi(h,t)$ from $\text{Cyl}(\mathcal{H} \times [0,1])$ and use the continuity equation in weak form, we assume term $\varphi(h,t)\nabla \cdot \mathcal{G}_t(h|z)$ disappears under integration. Thus we have

$$\begin{aligned}
\int_0^1 \int_{\mathcal{H}} \frac{\partial \varphi(h,t)}{\partial t} d\mu_t(h) dt &= -\int_0^1 \int_{\mathcal{H}} \int_z \langle \mathcal{G}_t(h|z), \nabla_h \varphi(h,t) \rangle d\mu_t(h|z) d\pi(z) dt \\
&= -\int_0^1 \int_{\mathcal{H}} \int_z \langle \mathcal{G}_t(h|z), \nabla_h \varphi(h,t) \rangle \frac{d\mu_t(h|z)}{d\mu_t(h)} d\mu_t(h) d\pi(z) dt \\
&= -\int_0^1 \int_{\mathcal{H}} \int_z \langle \mathcal{G}_t(h|z) \frac{d\mu_t(h|z)}{d\mu_t(h)}, \nabla_h \varphi(h,t) \rangle d\mu_t(h) d\pi(z) dt \\
&= -\int_0^1 \int_{\mathcal{H}} \int_z \langle \mathcal{G}_t(h|z) \frac{d\mu_t(\cdot|z)}{d\mu_t}(h) d\pi(z), \nabla_h \varphi(h,t) \rangle d\mu_t(h) dt \\
&= -\int_0^1 \int_{\mathcal{H}} \langle \int_z \mathcal{G}_t(h|z) \frac{d\mu_t(\cdot|z)}{d\mu_t}(h) d\pi(z), \nabla_h \varphi(h,t) \rangle d\mu_t(h) dt
\end{aligned}$$

On the other side, from Eq 29, we have

$$\int_0^1 \int_{\mathcal{H}} \frac{\partial \varphi(h,t)}{\partial t} d\mu_t(h) dt = -\int_0^1 \int_{\mathcal{H}} \langle \mathcal{G}_t(h), \nabla_h \varphi(h,t) \rangle d\mu_t(h) dt = 0, \quad \forall \varphi \in \text{Cyl}(\mathcal{H} \times [0,1])$$

Thus $\mathcal{G}_t(h) = \int_z \mathcal{G}_t(h|z) \frac{d\mu_t(\cdot|z)}{d\mu_t}(h) d\pi(z) = \int_{\mathcal{H}\times\mathcal{H}} \mathcal{G}_t(h|z) \frac{d\mu_t(\cdot|z)}{d\mu_t}(h) d\pi(z)$

H Derivation of Eq. 14

In this part, we show the detailed derivation of Eq. 14. In Flow Matching, the variable z is chosen as a single data point from the coupling $\pi(u_0, u_1)$ where $u_1 \sim q_1$, and $u_0 \sim q_0 = \mathcal{N}(\mathbf{0}, K(\{x_1, x_2, \ldots, x_n\}))$. Considering the class of Gaussian conditional probability paths

$$p_t(u_t|z) = \mathcal{N}(u_t|m_t(z), \sigma_t(z)^2 K(\{x_1, x_2, \ldots, x_n\})) \tag{31}$$

With conditional flow $\phi_t(u_t|z) = \sigma_t u_0 + m_t$. Specially, we choose $m_t = tu_1 + (1-t)u_0$ and $\sigma_t = \sigma$, where $\sigma > 0$ is a small constant. From Eq. 1 (or Theorem 3 of Lipman et al. [11]), a vector that defines the Gaussian conditional flow is :

$$\mathcal{G}_t(u_t|z) = \frac{\sigma_t'}{\sigma_t}(u_t - m_t) + m_t'(u_1) \tag{32}$$

Then we can derive a closed-form expression for both the conditional probability and corresponding vector field [2] by plug in μ_t and σ_t into Eq. 31 and Eq. 32

$$p_t(u_t|z) = \mathcal{N}(u_t|tu_1 + (1-t)u_0, \sigma^2 K(\{x_1, x_2, \ldots, x_n\})) \tag{33}$$

$$\mathcal{G}_t(u_t|u_1) = 0 + (u_1 - u_0) = u_1 - u_0 \tag{34}$$

Now, let's check the boundary conditions. At $t = 0$,

$$p_0(u_t|z) = \mathcal{N}(u_t|u_0, \sigma^2 K(\{x_1, x_2, \ldots, x_n\}) \xrightarrow{\sigma \to 0} \delta_{u_0} \tag{35}$$

At $t = 1$,

$$p_1(u_t|z) = \mathcal{N}(u_t|u_1, \sigma^2 K(\{x_1, x_2, \ldots, x_n\}) \xrightarrow{\sigma \to 0} \delta_{u_1} \tag{36}$$

From Eq. 9, we have $p_0(u_0) = \int p_0(u_t|z)\pi(z)dz = \int \delta_{u_0}\pi(u_0, u_1)du_0du_1 = q_0$ and $p_1(u_1) = \int p_1(u_t|z)\pi(z)dz = \int \delta_{u_1}\pi(u_0, u_1)du_0du_1 = q_1$, which show boundary conditions are satisfied.

I Proof of Proposition 3.1

Proposition 3.1. Given noisy observations $\{\widehat{u}(x_i)\}_{i=1}^n$, the posterior distribution is

$$\log \mathbb{P}\left(\{u(x_i)\}_{i=1}^m \middle| \{\widehat{u}(x_i)\}_{i=1}^n\right) = -\frac{\sum_{i=1}^n \|\widehat{u}(x_i) - u(x_i)\|^2}{2\sigma^2} + \log \mathbb{P}\left(\{u(x_i)\}_{i=1}^m\right) + C \tag{37}$$

Where the constant $C = -\frac{n}{2}\log(2\pi\sigma^2) - \log \mathbb{P}\left(\{\widehat{u}(x_i)\}_{i=1}^n\right)$.

Proof. With Bayes rule, we have:

$$\mathbb{P}\left(\{u(x_i)\}_{i=1}^m \middle| \{\widehat{u}(x_i)\}_{i=1}^n\right) = \frac{\mathbb{P}\left(\{\widehat{u}(x_i)\}_{i=1}^n \middle| \{u(x_i)\}_{i=1}^m\right) \cdot \mathbb{P}\left(\{u(x_i)\}_{i=1}^m\right)}{\mathbb{P}\left(\{\widehat{u}(x_i)\}_{i=1}^n\right)} \tag{38}$$

Taking the logarithm of Eq. 38, we have:

$$\log \mathbb{P}\left(\{u(x_i)\}_{i=1}^m \middle| \{\widehat{u}(x_i)\}_{i=1}^n\right) = \log \mathbb{P}\left(\{\widehat{u}(x_i)\}_{i=1}^n \middle| \{u(x_i)\}_{i=1}^m\right) + \log \mathbb{P}\left(\{u(x_i)\}_{i=1}^m\right) - \log \mathbb{P}\left(\{\widehat{u}(x_i)\}_{i=1}^n\right) \tag{39}$$

Given $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$ and $\{\epsilon_i\}_{i=1}^n$ is a multivariate Gaussian, then $\{\widehat{u}(x_i)\}_{i=1}^n \Big| \{u(x_i)\}_{i=1}^n$ is a shifted multivariate Gaussian with mean $\{u(x_i)\}_{i=1}^n$ translated from the original multivariate Gaussian $\{\epsilon_i\}_{i=1}^n$. Due to the translation invariance property of Gaussian distribution, We have :

$$\log \mathbb{P}\left(\{\widehat{u}(x_i)\}_{i=1}^n \middle| \{u(x_i)\}_{i=1}^n\right) = \log \mathbb{P}\left(\{\epsilon_i\}_{i=1}^n\right) = -\frac{\sum_{i=1}^n \|\widehat{u}(x_i) - u(x_i)\|^2}{2\sigma^2} - \frac{n}{2}\log(2\pi\sigma^2) \tag{40}$$

We notice $m > n$ and $\{\widehat{u}(x_i)\}_{i=1}^n$ only depends on $\{u(x_i)\}_{i=1}^n$, and doesn't depend on $\{u(x_i)\}_{i=n+1}^m$. Thus $\log \mathbb{P}\left(\{\widehat{u}(x_i)\}_{i=1}^n \middle| \{u(x_i)\}_{i=1}^m\right) = \log \mathbb{P}\left(\{\widehat{u}(x_i)\}_{i=1}^n \middle| \{u(x_i)\}_{i=1}^n\right)$.

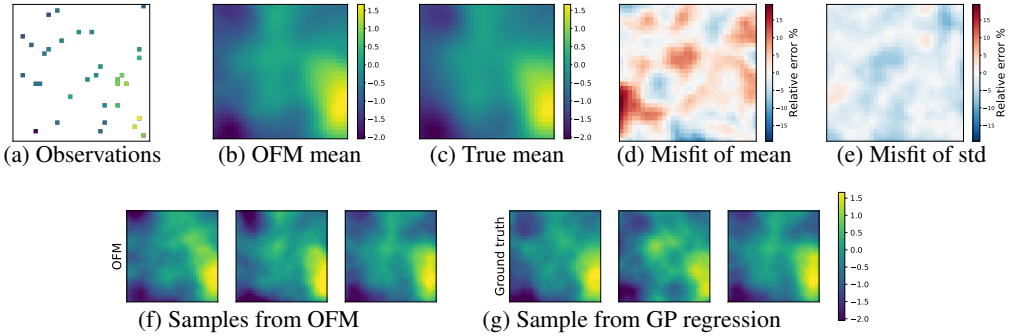
For evaluating $\log \mathbb{P}\left(\{u(x_i)\}_{i=1}^m\right)$, which is the second part on the right-hand side of Eq. 39, we can efficiently calculate it with the trace estimator. The third part on the right hand side of Eq. 39 ($\log \mathbb{P}\left(\{\widehat{u}(x_i)\}_{i=1}^n\right)$) represents the evidence and is constant. Thus the posterior distribution of Eq 39 can be simplified as:

$$\log \mathbb{P}\left(\{u(x_i)\}_{i=1}^m \middle| \{\widehat{u}(x_i)\}_{i=1}^n\right) = -\frac{\sum_{i=1}^n \|\widehat{u}(x_i) - u(x_i)\|^2}{2\sigma^2} + \log \mathbb{P}\left(\{u(x_i)\}_{i=1}^m\right) + C \tag{41}$$

Where the constant $C = -\frac{n}{2}\log(2\pi\sigma^2) - \log \mathbb{P}\left(\{\widehat{u}(x_i)\}_{i=1}^n\right)$. □

J Example of Posterior Samples

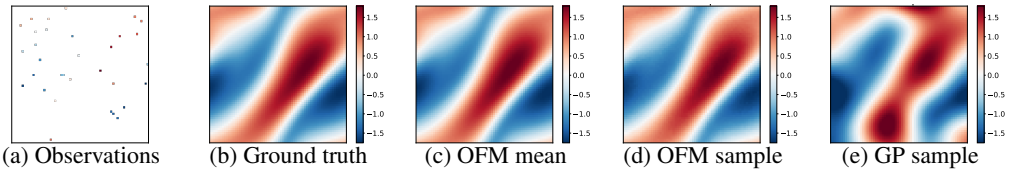
In this section, we initially present the regression result of OFM in another additional N-S scenario, as illustrated in Fig 7. Subsequently, we display more posterior samples used in the 2D regression examples. As depicted in Fig 9, 10, 11, OFM successfully generates realistic posterior samples that are consistent with the ground truth and demonstrate appropriate variability. In contrast, GP regression fails to produce explainable posterior samples.



(a) Observations (b) OFM mean (c) True mean (d) Misfit of mean (e) Misfit of std

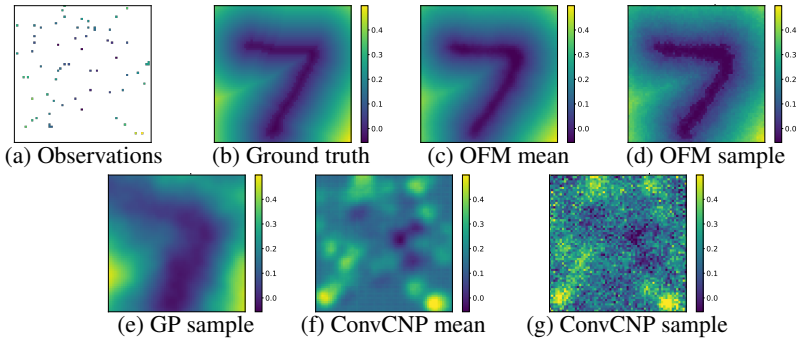
(f) Samples from OFM (g) Sample from GP regression

Figure 6: OFM regression on 2D GP data with resolution 32×32. (a) 32 random observations. (b) Predicted mean from OFM. (c) Ground truth mean from GP regression. (d) Misfit of the predicted mean. (e) Misfit of predicted standard deviation. (f) Predicted samples from OFM. (g) Predicted samples from GP regression.



(a) Observations (b) Ground truth (c) OFM mean (d) OFM sample (e) GP sample

Figure 7: OFM regression on Navier-Stokes functional data with resolution 64×64. (a) 32 random observations. (b) Ground truth sample (c) Predicted mean from OFM. (d) One posterior sample from OFM. (e) One posterior sample from best fitted GP.



(a) Observations (b) Ground truth (c) OFM mean (d) OFM sample

(e) GP sample (f) ConvCNP mean (g) ConvCNP sample

Figure 8: OFM regression on MNIST-SDF with resolution 64×64. (a) 64 random observations. (b) Ground truth sample. (c) Predicted mean from OFM. (d) One posterior sample from OFM. (e) One posterior sample from best fitted GP. (f) Predicted mean from ConvCNP. (g) One posterior sample from ConvCNP.

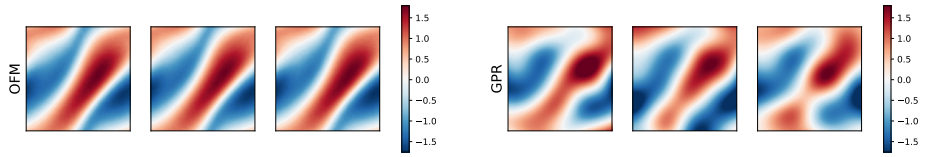


Figure 9: OFM regression on NS data. (**left**) Posterior samples from OFM. (**right**) Posterior samples from GP regression.

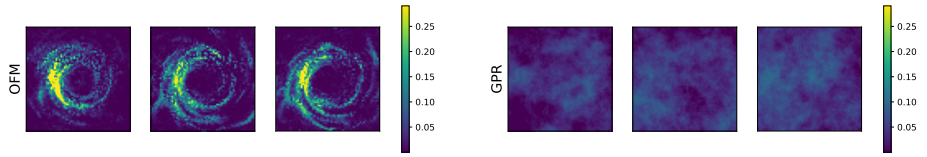


Figure 10: OFM regression on black hole data. (**left**) Posterior samples from OFM. (**right**) Posterior san

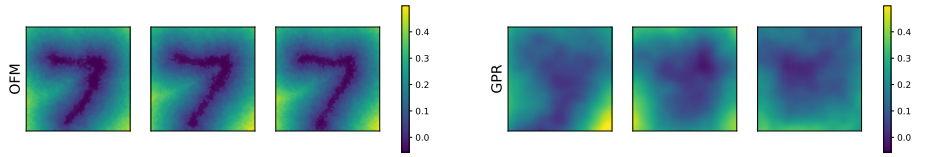


Figure 11: OFM regression on MNIST-SDF data. (**left**) Posterior samples from OFM. (**right**) Posterior samples from GP regression.

K Co-domain functional regression with OFM

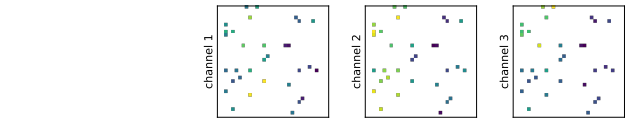
In this section, we expand our regression framework to accommodate co-domain settings, as many function datasets feature a co-domain dimension greater than one. For example, earthquake waveform data commonly include three directional components, leading to a three-dimensional co-domain. Similarly, the velocity field in fluid dynamics usually features three directional components, also resulting in a dimension of co-domain of three.

We illustrate this extension through a 2D GP example with a co-domain of 3 (channel dimension of 3). In learning the prior, we define the reference measure (ν_0) as a joint measure (Wiener measure) of three identical but independent Gaussian measures while the target measure (ν_1) is another Wiener measure. We keep all other parameters unchanged as those described in the 2D GP regression tasks, with the only modification being an increase in the channel dimension from one to three. After training the prior (training detail provided in Appendix M), and provided 32 random observations across the three channels at co-locations, we then perform regression with OFM across these channels jointly. As demonstrated in Fig 12, OFM accurately estimate the mean and uncertainty across three channels.

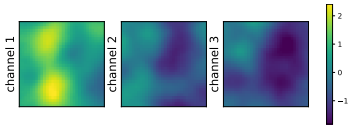
L Posterior sampling with Stochastic Gradient Langevin Dynamics

In this section, we describe how to sample from posterior distribution with SGLD. We denote logarithmic posterior distribution (Eq. 41) as $\log \mathbb{P}_\theta$ and denote a set of posterior samples as $\{u_\theta^t\}_{t=1}^N$, where each u_θ^t is defined on a collection of point $\{x_i\}_{i=1}^m$.

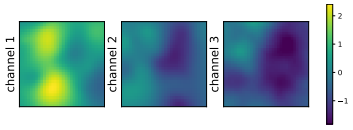
By following the standard SGLD pipeline as described by Welling and Teh [4], we can obtain a set of N posterior samples $\{u_\theta^t\}_{t=1}^N$. However, SGLD is known to be sensitive to the choice of regression parameters and can become trapped in local minima, leading to convergence issues, especially in regions of high curvature [51]. To mitigate these challenges, Shi et al. [1] proposed that within an invertible framework, drawing a posterior sample u_θ^t is equivalent to drawing a sample a_θ^t in Gaussian space, since u_θ^t uniquely defines a_θ^t and vice versa. This approach can stabilize the posterior sampling process and is less sensitive to the regression parameters due to the inherent smoothness of the Gaussian process. Additionally, Shi et al. [1] suggests starting from maximum a *posteriori*



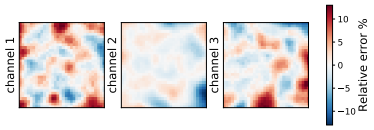
(a) Partial observations



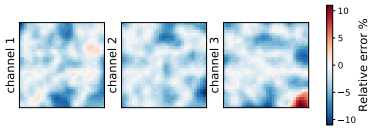
(b) OFM mean



(c) True mean



(d) Misfit of mean



(e) Misfit of standard deviation

Figure 12: OFM regression on co-domain GP data with resolution 32x32. (a) 32 random observations at co-locations. (b) Predicted mean from OFM. (c) Ground truth mean from GP regression. (d) Misfit of the predicted mean. (e) Misfit of predicted standard deviation.

(MAP) estimate of a_θ^t, denoted as $\overline{a_\theta}$, which can reduces the number of burn-in terations needed in SGLD. We adopt the same sampling strategy and the algorithm is reported in Algorithm 1

When the size of observations or context points ($\{\widehat{u}(x_i)\}_{i=1}^n$) is 0, sampling from the posterior degrades to sampling from the prior, the results of which are presented in the subsequent section.

Algorithm 1 Posterior sampling with SGLD

Input and Parameters: Logarithmic posterior distribution $\log \mathbb{P}_\theta$, temperature T, learning rate η_t, MAP $\overline{a}_\theta$, burn-in iteration b, sampling iteration t_N, total iteration N.

Initialization: $a_\theta^0 = \overline{a}_\theta$
for $t = 0, 1, 2, \ldots, N$ **do**
  Compute gradient of the posterior: $\nabla_{a_\theta} \log \mathbb{P}_\theta$
  Update a_θ^{t+1}: $a_\theta^{t+1} = a_\theta^t + \frac{\eta_t}{2} \nabla \log \mathbb{P}_\theta + \sqrt{\eta_t T} \mathcal{N}(0, I)$
  if $t \geq b$ **then**
    Every t_N iterations: obtain new sample a_θ^{t+1}, and corresponding u_θ^{t+1}
  end if
end for

M Prior learning with OFM

We now elaborate on the prior learning process and the corresponding performance evaluation. As shown in Algorithm 2, the training dataset is sampled from the unknown data measure ν_1. Concretely, the training dataset consists of M discretized functions $\{u_i|D_i\}_{i=1}^M$, where $u_i|D_i$ denotes a discretized observation of the u_i function.

In practice, to simplify dataset preparation, one often uses the same discretization grid D_i for all function samples, e.g. $D_i = \{x_1, \cdots, x_n\}$ regardless of the sample index "i". For the consistency of notions, let h_0 represents a batch of i.i.d discretized functions sampled from the training dataset (equivalently, sampled from ν_1). Next, the reference Gaussian process $\nu_0 = \mathcal{N}(m_0, C_0)$ is known and determined by the user. With a slight abuse of notation, We choose to use notation h_0, h_1 for consistency purpose, in other parts of this paper, discretized h_0, h_1 is replaced with a, u respectively.

For specific experiments setting, we employ Matern kernel to construct the reference GP and to prepare training datasets for 1D GP, 2D GP, and 1D TGP. We have set the kernel length $l = 0.01$ with a smoothness factor $\zeta = 0.5$ for all reference GPs. OFM maps the GP samples from reference GPs to data samples and is resolution-invariant, which means OFM can be trained with functions at any resolution and evaluated at any resolution.

Algorithm 2 Learning a prior

Input: Reference Gaussian process $\nu_0 = \mathcal{N}(m_0, C_0)$, data measure ν_1, batch size b, small constant σ_{min}, discretized domain $D = \{x_1, \cdots x_n\}$

while Training **do**
  $h_0 \sim \nu_0; \quad h_1 \sim \nu_1$ # sample functions of size b i.i.d from the measures on D
  $\pi \leftarrow \text{OT}(h_0, h_1)$ # mini-batch optimal transport plan
  $(h_0, h_1) \sim \pi$
  $t \sim \mathcal{U}(0, 1)$
  $\mu_t \leftarrow t\, h_1 + (1 - t)\, h_0$
  $h_t \sim \mathcal{N}(\mu_t, \sigma_{\text{min}}^2 C_0)$
  $\mathcal{L}^{\dagger}_{\text{CFM}}(\theta) \leftarrow \left\| \mathcal{G}_\theta(t, x) - (h_1 - h_0) \right\|^2$
  $\theta \leftarrow \text{Update}\big(\theta, \nabla_\theta \mathcal{L}^{\dagger}_{\text{CFM}}(\theta)\big)$
end while
return $\mathcal{G}_\theta$

1D GP dataset. We choose $l = 0.3$ and $\zeta = 1.5$ and generate $20,000$ training samples on domain $[0, 1]$ with a fixed resolution of 256. We use autocovariance and histogram of point-wise value as metrics for evaluation. We evaluate OFM at several different resolutions shown Fig 13, 14, 15, which demonstrate OFM's excellent capability to learn the function prior.

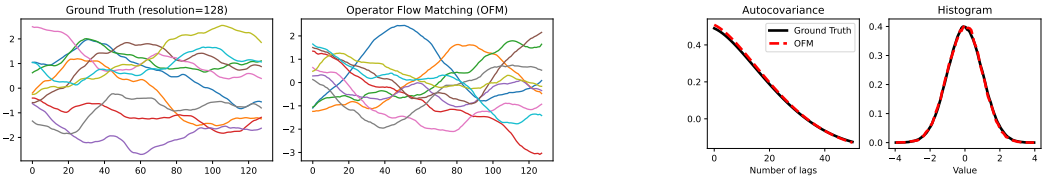


Figure 13: OFM for 1D GP prior learning, evaluated at resolution=128. (**left two**) Random samples from ground truth and generated by OFM. (**right two**) Autocovariance and histogram comparison

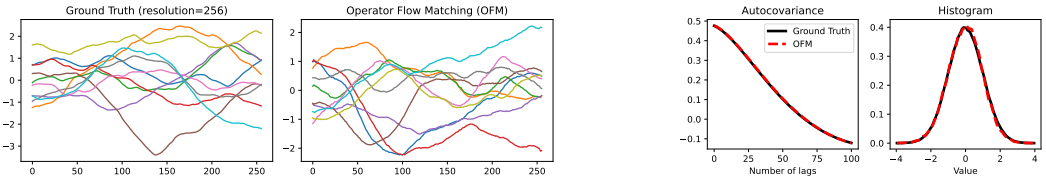


Figure 14: OFM for 1D GP prior learning, evaluated at resolution=256. (**left two**) Random samples from ground truth and generated by OFM. (**right two**) Autocovariance and histogram comparison

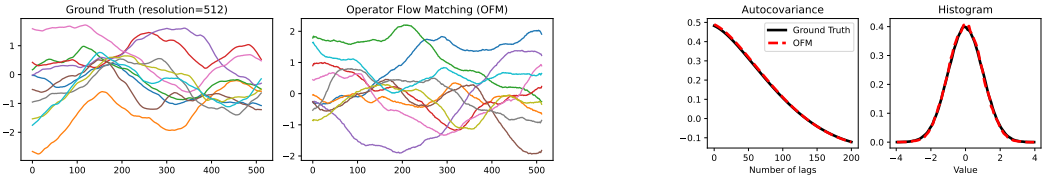


Figure 15: OFM for 1D GP prior learning, evaluated at resolution=512. (**left two**) Random samples from ground truth and generated by OFM. (**right two**) Autocovariance and histogram comparison

1D TGP dataset. We choose $l = 0.3$ and $\zeta = 1.5$ and generating $20,000$ training samples on domain $[0, 1]$ with a fixed resolution of 256. We set $[-1.2, 1.2]$ for the bounds. Results provided in Fig 16, 17, 18.

2D Naiver-Stokes, Black hole, MNIST-SDF datasets. All the following 2D datasets are defined on domain $[0, 1] \times [0, 1]$ and have a resolution of 64×64. We collected a 2D Navier-Stokes dataset

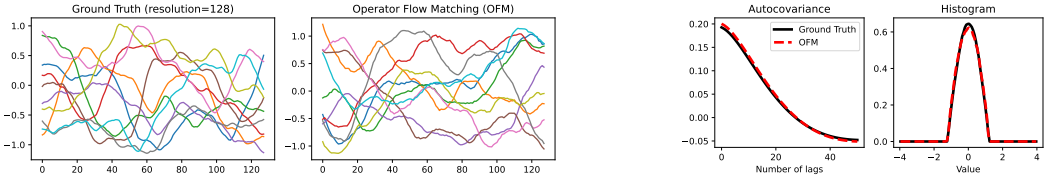


Figure 16: OFM for 1D TGP prior learning, evaluated at resolution=128. (**left two**) Random samples from ground truth and generated by OFM. (**right two**) Autocovariance and histogram comparison

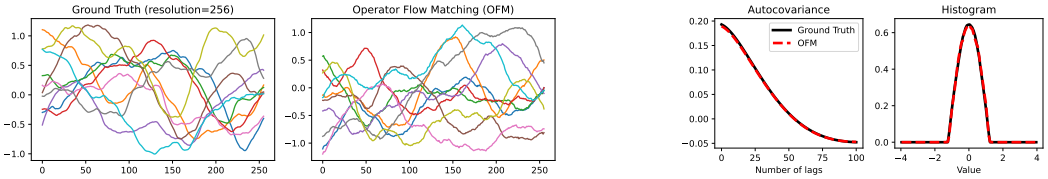


Figure 17: OFM for 1D TGP prior learning, evaluated at resolution=256. (**left two**) Random samples from ground truth and generated by OFM. (**right two**) Autocovariance and histogram comparison

consisting of 20000 samples, with viscosity $= 1e-4$. The results, including zero-shot super-resolution, are provided in Fig 19, 20. The learning of Black hole dataset, generated using expensive Monte Carlo method, is detailed in Fig 21, 22. Additionally, we trained OFM on $20,000$ MNIST-SDF samples, the outcomes are illustrated in Fig 23, 24.

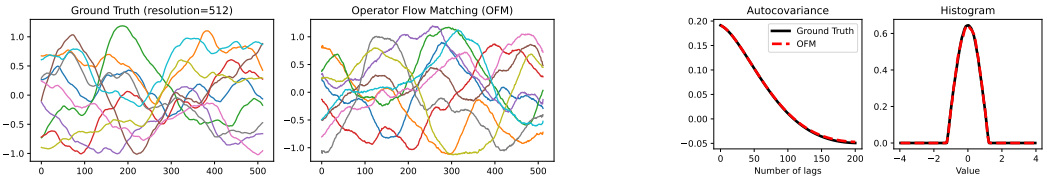


Figure 18: OFM for 1D TGP prior learning, evaluated at resolution=512. (**left two**) Random samples from ground truth and generated by OFM. (**right two**) Autocovariance and histogram comparison

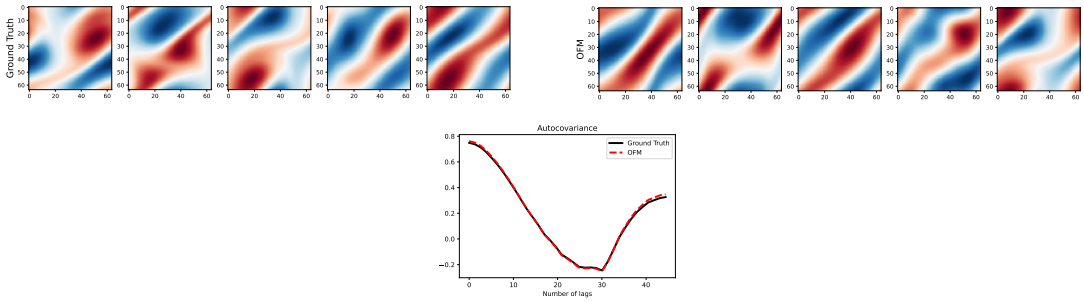


Figure 19: OFM for 2D N-S prior learning, evaluated at resolution=64×64. (**top left**) Random samples from ground truth. (**top right**) Random samples generated by OFM. (**bottom**) Autocovariance comparison

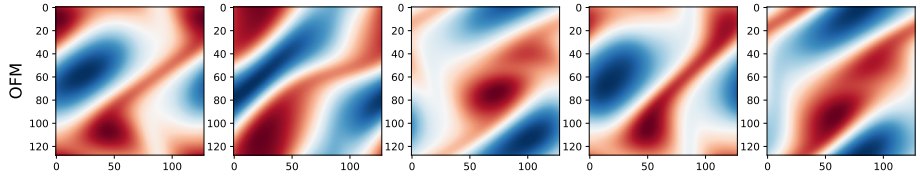


Figure 20: OFM for 2D N-S prior learning, evaluated at 128×128 resolution (zero-shot super-resolution)

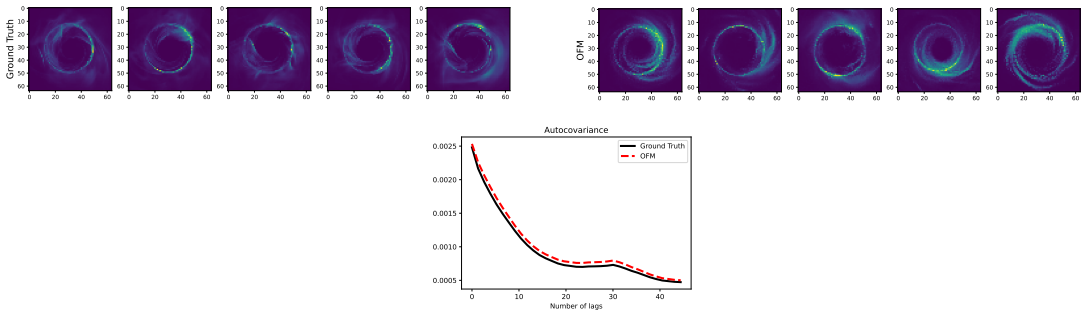


Figure 21: OFM for 2D black hole prior learning, evaluated at resolution=64. (**top left**) Random samples from ground truth. (**top right**) Random samples generated by OFM. (**bottom**) Autocovariance comparison

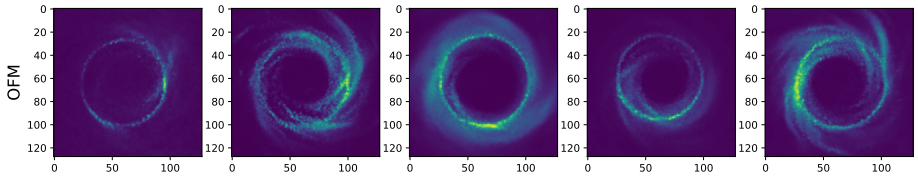


Figure 22: OFM for 2D black hole prior learning, evaluated at 128×128 resolution (zero-shot super-resolution)

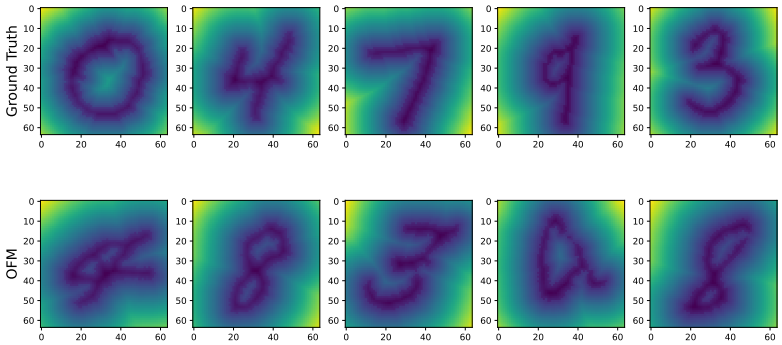


Figure 23: OFM for 2D MNIST-SDF prior learning, evaluated at 64×64 resolution. (**top**) Random samples from ground truth. (**bottom**) Random samples generated by OFM.

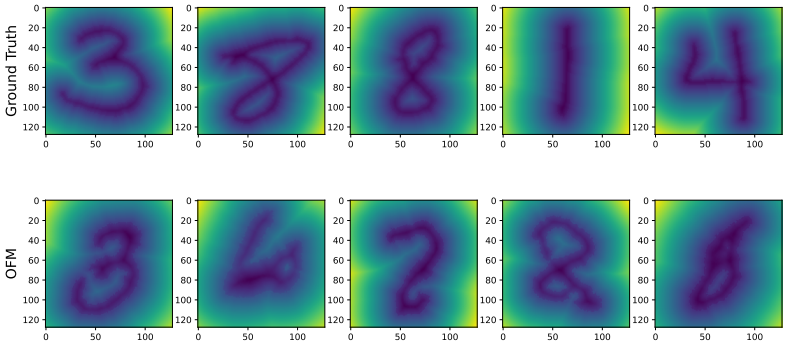


Figure 24: OFM for 2D MNIST-SDF prior learning, evaluated at 128×128 resolution. (**top**) Random samples from ground truth. (**bottom**) Random samples generated by OFM.

N Additional results of 1D GP regression with more complex kernels

In this section, we extend the 1D GP experiments by introducing more complex kernels for both data generation and regression, while holding all other settings identical to those used with the Matérn kernel. Specifically, we report results for the non-stationary Gibbs kernel and the Rational Quadratic (RQ) kernel.

For the Gibbs kernel, we use an input-dependent length-scale

$$\ell(x) = \ell_0 + \ell_1 x, \qquad x \in [0, 1],$$

which induces the covariance

$$k(x, x') \;=\; \sigma^2 \sqrt{\frac{2\,\ell(x)\,\ell(x')}{\ell(x)^2 + \ell(x')^2}}\; \exp\!\left(-\frac{(x - x')^2}{\ell(x)^2 + \ell(x')^2}\right).$$

In our setup we take $\ell_0 = 0.05$, $\ell_1 = 0.25$, and $\sigma = 1.0$.

For the RQ kernel, we use `sklearn.gaussian_process.kernels.RationalQuadratic` with length-scale 0.15. As shown in Table 2, OFM consistently outperforms all baselines on these tasks.

Table 2: Comparison of OFM with baseline models on 1D GP with Gibbs kernel and RQK. Best performance in bold.

Dataset →	1D GP-Gibbs		1D GP-RQK	
Algorithm ↓ Metric →	SMSE	MSLL	SMSE	MSLL
NP	$4.0 \cdot 10^{-1}$	$7.1 \cdot 10^{-1}$	$5.8 \cdot 10^{-1}$	$2.2 \cdot 10^{0}$
ANP	$3.5 \cdot 10^{-1}$	$6.2 \cdot 10^{-1}$	$2.7 \cdot 10^{-1}$	$2.1 \cdot 10^{0}$
ConvCNP	$3.4 \cdot 10^{-1}$	$1.1 \cdot 10^{-1}$	$2.2 \cdot 10^{-1}$	$6.9 \cdot 10^{-1}$
DGP	$4.3 \cdot 10^{-1}$	$8.6 \cdot 10^{-1}$	$2.4 \cdot 10^{-1}$	$1.2 \cdot 10^{0}$
DSPP	$4.2 \cdot 10^{-1}$	$7.1 \cdot 10^{-1}$	$2.5 \cdot 10^{-1}$	$4.2 \cdot 10^{-1}$
OpFlow	$3.1 \cdot 10^{-1}$	$6.9 \cdot 10^{-1}$	$2.9 \cdot 10^{-1}$	$5.0 \cdot 10^{-1}$
OFM(Ours)	$\mathbf{2.9 \cdot 10^{-1}}$	$\mathbf{8.7 \cdot 10^{-2}}$	$\mathbf{2.1 \cdot 10^{-1}}$	$\mathbf{9.4 \cdot 10^{-2}}$

O Details of experimental setup

In this section, we outline the details of experiments setup used in this paper. Since regression with OFM requires learning the prior first, we list the parameters used for learning the prior and regression separately. We employ FNO as the backbone, implemented using `neuraloperator` library [14]. All time reported in the subsequent tables are based on one computations performed using a single NVIDIA RTX A6000 (48 GB) graphics card. In all experiments, we use the `dopri5` ODE solver provided by `torchdiffeq` Chen et al. [52] with `atol=1e-5` and `rtol=1e-5`. Detailed posterior sampling algorithm is provided in Appendix L.

Table 3 details the parameters used for training the prior. For instance, in the 1D GP prior learning experiment, the dataset consists of 20,000 samples, each with a co-domain dimension (or channel) of one. The batch size is set at 1024, and the model is trained over 500 epochs. The total training time is about 0.76 hours, and the size of the trained model is 37.1 megabytes.

Tables 4, 5, and 6 detail the parameters for SGLD sampling as described in Algorithm 1. For example, in the 1D GP regression experiment, the regression takes 40,000 iterations with a burn-in phase of 3,000 iterations. Posterior samples are collected every 10 iterations. The temperature for the injected noise during the gradient update is set at 1, and the learning rate decays exponentially from 0.005 to 0.004 (defined in Algorithm 1). We average 32 runs with the Hutchinson trace estimator to evaluate the likelihood, utilizing GPU parallel computing. The noise level, as specified in Equation 41, is 0.01 in this regression task. Then given 6 random observations, we ask for the posterior samples across 128 points. The GPU memory usage for the regression task is 4 gigabytes, with the total runtime to 4.91 hours.

Table 3: Parameters used in experiments of prior learning

Datasets	Size of Dataset	Channels	Batch Size	Epochs	Training Time	Model Size
1D GP	$2 \cdot 10^4$	1	1024	$5 \cdot 10^2$	0.76 h	37.1 MB
1D TGP	$2 \cdot 10^4$	1	1024	$5 \cdot 10^2$	1.24 h	37.1 MB
2D GP	$2 \cdot 10^4$	1	256	$5 \cdot 10^2$	1.14 h	76 MB
2D co-domain GP	$2 \cdot 10^4$	3	256	$5 \cdot 10^2$	1.01 h	76 MB
2D N-S	$2 \cdot 10^4$	1	256	$5 \cdot 10^2$	3.79 h	286 MB
2D Black hole	$1.2 \cdot 10^4$	1	256	$5 \cdot 10^2$	2.28 h	286 MB
2D MNIST-SDF	$2 \cdot 10^4$	1	256	$5 \cdot 10^2$	8.31 h	286 MB

Table 4: Parameters used in regression experiments - Part A

Datasets	Total Iteration	Burn-in Iteration	Sampling Iterations	Temperature of Noise
1D GP	$4 \cdot 10^4$	$3 \cdot 10^3$	10	1
1D TGP	$4 \cdot 10^4$	$3 \cdot 10^3$	10	1
2D GP	$2 \cdot 10^4$	$3 \cdot 10^3$	10	1
2D co-domain GP	$2 \cdot 10^4$	$3 \cdot 10^3$	10	1
2D N-S	$2 \cdot 10^4$	$3 \cdot 10^3$	10	1
2D Black hole	$2 \cdot 10^4$	$3 \cdot 10^3$	10	1
2D MNIST-SDF	$2 \cdot 10^4$	$3 \cdot 10^3$	10	1

P Ablation and scaling studies for mini-batch optimal transport and Hutchinson trace estimator

We first present an ablation study of the optimal transport plan. In this study, we revisit prior learning on the N-S dataset by training two models: one using independent coupling and the other employing a mini-batch optimal transport plan. Both models were trained with a batch size of 64. For evaluation, we compare the mean squared error (MSE) of density, autocovariance and spectral characteristics between 1,000 real and generated samples. Additionally, we report the convergent training loss (squared L^2 loss) and the number of function evaluations (NFE) required for sampling with adaptive ODE solver (dopri5). As shown in Table 7, the mini-batch OT plan outperforms the independent coupling approach in terms of pointwise accuracy, spectral fidelity, and convergent L^2 loss, while also requiring fewer NFE and enabling faster sampling. Our findings indicate that as the mini-batch size increases, the model learns the prior with reduced error. This improvement may be attributed to the mini-batch optimal transport plan approaching the true optimal transport plan with larger batch sizes.

Next, we explore the variance of the Hutchinson trace estimator in likelihood estimation with a scaling experiment. For this experiment, we use a 1D GP example with parameters described in Section M and a resolution of 128. We randomly draw a GP sample from the prior and evaluate the log likelihood by integrating the divergence as shown in Eq. 15, while also estimating it using the Hutchinson trace estimator from Eq. 16. In this scaling experiment, the number of noise sample (n_{noise}) for the Hutchinson estimator is set to $(4, 8, 16, 32, 64, 128)$. For each n_{noise}, we repeat the experiment 100 times and report the mean and standard deviation of the predicted likelihood. The exact likelihood is computed by directly integrating the trace of the Jacobian. We repeat this procedure for 5 different random 1D GP samples. As reported in Table 8, the standard deviation of the Hutchinson trace estimator decreases rapidly as n_{noise} increases, and the predicted means always align closely with the ground truth.

Furthermore, even at smaller n_{noise} values, where a relative larger variance is expected, the performance of the posterior sampling appears robust (see Table 5). We hypothesize that this is due to stochastic nature of posterior sampling algorithm (SGLD), which requires injected perturbations, renders its performance relatively insensitive to the choice of n_{noise}.

Table 5: Parameters used in regression experiments - Part B

Datasets	Initial Learning Rate	End Learning Rate	Hutchinson Samples	Noise Level
1D GP	$5 \cdot 10^{-3}$	$4 \cdot 10^{-3}$	32	$1 \cdot 10^{-2}$
1D TGP	$5 \cdot 10^{-3}$	$4 \cdot 10^{-3}$	32	$1 \cdot 10^{-3}$
2D GP	$1 \cdot 10^{-3}$	$8 \cdot 10^{-4}$	32	$1 \cdot 10^{-2}$
2D co-domain GP	$1 \cdot 10^{-3}$	$8 \cdot 10^{-4}$	16	$1 \cdot 10^{-2}$
2D N-S	$3 \cdot 10^{-3}$	$2 \cdot 10^{-3}$	8	$1 \cdot 10^{-3}$
2D Black hole	$5 \cdot 10^{-3}$	$4 \cdot 10^{-3}$	8	$1 \cdot 10^{-3}$
2D MNIST-SDF	$5 \cdot 10^{-3}$	$4 \cdot 10^{-3}$	8	$1 \cdot 10^{-3}$

Table 6: Parameters used in regression experiment - Part C

Datasets	Number of Observations	Inquired Grids	GPU Memory	Running Time
1D GP	6	128	4 GB	4.91 h
1D TGP	3	128	4 GB	5.42 h
2D GP	32	32×32	22 GB	9.70 h
2D co-domain GP	32	32×32	31 GB	5.05 h
2D N-S	32	64×64	44 GB	13.65 h
2D Black hole	32	64×64	44 GB	13.37 h
2D MNIST-SDF	64	64×64	44 GB	9.41 h

Table 7: Scaling study for the size of mini-batch given optimal transport plan, best performance in bold

Metrics	Density-MSE	Autocovariance-MSE	Spectra-MSE	Convergent L^2 Loss	NFE
Independent	$3.4 \cdot 10^{-5}$	$7.4 \cdot 10^{-5}$	$1.3 \cdot 10^{1}$	$5.1 \cdot 10^{-2}$	168
mini-batch = 32	$3.0 \cdot 10^{-5}$	$1.3 \cdot 10^{-4}$	$5.0 \cdot 10^{0}$	$2.3 \cdot 10^{-2}$	**141**
mini-batch = 64	$\mathbf{9.8 \cdot 10^{-6}}$	$9.8 \cdot 10^{-5}$	$6.3 \cdot 10^{0}$	$1.9 \cdot 10^{-2}$	153
mini-batch = 128	$2.4 \cdot 10^{-5}$	$\mathbf{1.7 \cdot 10^{-5}}$	$\mathbf{3.6 \cdot 10^{0}}$	$\mathbf{1.5 \cdot 10^{-2}}$	182

Table 8: Scaling study for the number of noise samples of Hutchinson trace estimator

	$n_{\text{noise}} = 4$	$n_{\text{noise}} = 8$	$n_{\text{noise}} = 16$	$n_{\text{noise}} = 32$	$n_{\text{noise}} = 64$	$n_{\text{noise}} = 128$	exact
sample 1	-481.7 ± 9.4	-482.9 ± 7.7	-480.6 ± 4.9	-481.4 ± 3.8	-481.1 ± 2.5	-480.8 ± 1.8	-482.7
sample 2	-482.8 ± 9.8	-481.8 ± 6.7	-481.3 ± 4.9	-480.8 ± 3.5	-481.1 ± 2.7	-481.2 ± 1.7	-482.0
sample 3	-479.7 ± 10.2	-479.7 ± 7.6	-478.2 ± 4.7	-478.6 ± 3.4	-478.8 ± 2.6	-478.8 ± 1.7	-479.1
sample 4	-476.9 ± 10.0	-477.0 ± 6.6	-477.9 ± 5.6	-478.5 ± 4.0	-478.1 ± 2.7	-478.0 ± 1.8	-477.4
sample 5	-479.4 ± 10.7	-479.2 ± 6.6	-479.2 ± 5.3	-479.4 ± 3.6	-479.6 ± 2.8	-479.3 ± 2.0	-478.5

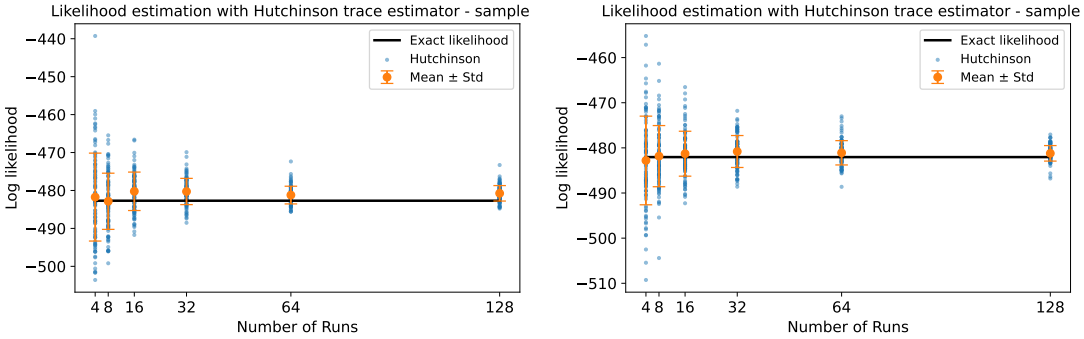


Figure 25: Log likelihood by the integration of divergence, (**left**) plot for first GP sample. (**right**) plot for second GP sample

Q Detailed analysis of OFM and comparison with existing methods

In this section, we elaborate the connection and difference with pervious work, highlight contributions and potential limitations of our work. The regression with OFM involves a two-steps process: (i) learning a prior on function space, and (ii) sampling from the posterior given observations. Consequently, the OFM framework has connections with both generative models on function space and the models developed for functional regression. In the following, we provide a comprehensive comparative analysis with related models and baselines, including operator flow (OPFLOW) [1], conditional optimal transport flow matching (COT-FM) [45], conditional models (NPs) [7, 34]

Comparison with OPFLOW. OPFLOW introduces invertible neural operators, which generalizes RealNVP [53] to function space and maps any collection of points sampled from a GP to a new collection of points in the data space, using the maximum likelihood principle [1]. This method captures the likelihood of any collection of point consistently as the resolution increases and allows for UFR using SGLD. Despite these advantages, the requirement for an invertible neural operator brings training and expressiveness challenges. To be specific, OPFLOW failed on all non-GP regression tasks in this paper because the prior learning stage suffered from mode collapse during training. On the contrary, OFM adopts a simulation-free ODE framework for prior learning, which offers enhanced expressiveness and ensures training stability through a simple regression objective while avoiding using the invertible neural operator. In addition, OFM proposes a non-trivial extension of UFR to the simulation-free ODE framework. These improvements render OFM a more practical solution for challenging functional regression tasks.

Comparison with COT-FM. COT-FM [45] proposes a conditional generalization of Benamou-Brenier Theorem [54], formulating a conditional optimal transport plan that applicable for both Euclidean and Hilbert space. In contrast, OFM employs an unconditional optimal transport plan in Hilbert space based on dynamic Kantorovich formulation, which is initially generalized for unbalanced optimal transport [37]. The advantage of COT-FM lies in its ability to flexibly incorporate specific conditions tailored for conditional generative tasks. However, COT-FM is not suitable for functional regression tasks due to: (i) COT-FM is contingent upon both the reference and target being influenced by conditions, and the vector field learnt is triangular, designed to transport jointly the coupling of a reference measure and a condition measure. In UFR setting, the learnt prior is required to be unconditioned, (ii) the coupling with condition measure typically prevents inducing valid stochastic process, even when the reference measure is a Gaussian measure, (iii) cannot provide point evaluation of probability density. Last, We should notice, the development of OFM is different and independent of COT-FM, the former with a focus on stochastic process learning and Bayesian functional regression.

Comparison with conditional models. NPs were developed to address the computational and restrictive prior challenges of Gaussian Processes, utilizing neural networks for efficiency [7]. However, several recent studies have discussed the drawbacks in the formulation of NPs, raising concerns that NPs might not learn the underlying function distribution [1, 22, 55].

Notably, NPs treats the point cloud data as a set of values, ignoring the metric space of the data [55]. This can lead to misinterpretations of a function sampled at different resolutions as distinct functions (Appendix A.1 of [22]). Furthermore, NPs rely on encoding input data into finite-dimensional, Gaussian-distributed latent variables before projecting these into an infinite-dimensional space. This process tends to lose consistency at higher resolutions. Moreover, the Bayesian regression framework underpinning NPs focuses on point sets rather than the functions themselves, leading to a dilution of prior information with increasing data points.

In recent study, diffusion-based variants of NPs (NDP) [34], was proposed to leverage the expressiveness of diffusion models [24, 25]. Nonetheless, the formulation of NDP does not address the aforementioned issues of NPs and introduces two more problems: (i) NDP fails to induce a valid stochastic process as it does not satisfy the marginal consistency criterion required by Kolmogorov Extension Theorem [39], and (ii) it relies on uncorrelated Gaussian noise for denoising, which is not applicable in function spaces [26]. Oppositely, OFM establishes a more theoretically sound framework by rigorously defining learning within function spaces. Additionally, Bayesian functional regression within the OFM framework adheres to valid stochastic processes, offering a robust and theoretically grounded solution. Last, we provide a high-level comparison of OFM and NP as shown in Table 9.

Table 9: High-level comparison of OFM and NP.

Aspect	**OFM**	**Neural Processes (NP)**
Modeling target	Global *prior* over stochastic processes with a valid joint density.	Amortized *conditional* model without explicit tractable joint density over full functions.
Arbitrary queries	Set-size / location agnostic; evaluate on any grid or point set.	Set-size / location agnostic; trained with random context/target splits. Performance may degrade under extreme sparsity or distribution shift
Uncertainty & likelihoods	Tractable log-densities and posterior sampling enable principled Bayesian inference.	Predictive densities are available and trained via conditional likelihood; no closed-form function-level likelihood is typically specified. Uncertainty reflects model/context coverage and may be miscalibrated.
Structure & extrapolation	Captures global geometry and non-stationarity via the learned flow.	Learns inductive biases from data through context summaries; strong interpolation, but extrapolation and long-range generalization are not guaranteed and depend on training distribution.

Limitations. Despite these advances, the current regression framework with OFM is primarily limited to low-dimensional data (1D and 2D in this study). This limitation stems from the challenges associated with learning operators for functions defined on high-dimensional domains—an area that remains underdeveloped both computationally and in terms of dataset availability [15]. Additionally, while the time complexity for regression with OFM is $\mathcal{O}(m^2)$, the incorporation of additional components significantly increases its computational resource requirements compared to classical GP regression.

There are several potential paths to mitigate the computation concerns: (i) Carefully adjusting the step size (η_t) and temperature (T) in the SGLD algorithm to prevent overly large gradient updates. (ii) Running the trace estimator on multiple GPUs, use mixed-precision FNO [43], or adopt efficient neural operators [56]. (iii) Using more efficient ODE solvers. Our current implementation uses a high-order ODE solver (dopri5) that requires over 100 steps for sampling. Switching to a more modern and efficient solver, such as the DPM-Solver [57], could potentially reduce the number of evaluations, freeing up significant GPU resources.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction succinctly outline the paper's claims and contributions, which are fully supported by the theoretical analysis and experimental results.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We also highlight the limitations in the main text and add explain into detail in appendix Q

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: We provide detailed explanations, derivations, and proofs for all theoretical results (see Appendix C, D, and F).

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide the detailed experimental setup in Appendix M, L, and O. We also include code with additional supplementary examples, log files, and figures to ensure that all results are fully reproducible.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The code is provided as supplementary material, along with example scripts, log files, and figures to ensure full reproducibility of our results.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: See Appendix O

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Following the convention in functional regression, we assess posterior learning using SMLL and MSME for GP examples. For non-GP tasks, we also provide visual checks by plotting the predicted mean alongside posterior samples.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See the Appendix O

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The authors do not foresee any ethical implications to this work.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: This work is not directly tied to any harmful applications.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This is not applicable

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite each reused code snippet and dataset with its original publication.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: We do not release new assets

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: We do not do crowdsourcing nor research with human subjects

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: We do not do crowdsourcing nor research with human subjects

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: This work is not related to LLM

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.