FLOW-ADAPTIVE NEURAL PROCESSES WITH MULTI-SCALE SPATIAL BIAS FOR IRREGULAR INPUTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning expressive function generation from irregular observations is challenging. Irregular observations can be random in available locations, order, and receiving numbers. To handle this problem, we introduce the *Structured Flow Process* (SFP), a novel generative neural process specifically designed to learn function priors through flow matching over finite-dimensional marginals and achieve efficient posterior inference. Our key innovation is *Flow-adaptive Multihead Attention*, which introduces multiscale spatial bias and flow-time conditioning into the velocity model to capture evolving correlations along the generative flow. SFP supports both prior modeling and posterior inference on irregular input sets and preserves the desired permutation invariance of the induced distributions. Empirically, SFP achieves strong performance across diverse prior-learning and posterior-inference tasks among synthetic stochastic processes, physical dynamical systems, and real geophysical fields. We further evaluate SFP on downstream applications, including tabular regression and Bayesian optimization with inputs of up to eight dimensions, demonstrating its scalability and broader applicability.

1 INTRODUCTION

Learning expressive function generation is important for data-efficient machine learning and scientific applications. A central challenge of this study is to model coherent correlations among function values across the input domain (Bishop & Nasrabadi, 2006). This becomes difficult with sparse and irregularly sampled observations, which are broadly included in point clouds data, scattered sensor data, and irregular time series. Unlike structured data such as images, irregular observations lack the organized structural information, making underlying correlations harder to identify and consistently represent across the varying, sparse and unordered input sets.

Gaussian processes (GPs) earlier provide a principled framework for this challenge in defining priors over functions and performing posterior inference at arbitrary input locations (Williams & Rasmussen, 2006). To improve scalability, Neural Processes (NPs), building on the function perspective of GPs, learn an amortized conditional model that maps a context set of observed input-output pairs to posterior distributions over target locations (Garnelo et al., 2018a;b). However, in standard NPs, the function prior is only implicitly represented through the learned conditional inference procedure. Their ability to represent function correlations is closely related to the prescribed context-target splits used during training, making the learned inference procedure sensitive to the context and target set-size distributions (Garnelo et al., 2018a; Le et al., 2018; Rochussen & Fortuin, 2025). Moreover, standard NPs typically perform posterior prediction via a single-pass conditional decoder, which restricts the learning and representation of complex posterior uncertainty (Dutordoir et al., 2023; Hamad & Rosenbaum, 2025). Above limitations motivate us to explore more expressive generative mechanisms for standard NPs.

Recent advances incorporate continuous generative models, mainly diffusion models (Song et al., 2021; Ho et al., 2020) and flow models (Lipman et al., 2023), into the NP framework (Dutordoir et al., 2023; Mathieu et al., 2023; Biggs & Willis, 2026). These methods model function distributions through the joint distribution of function values on finite input sets, with the learned prior characterized by correlations across input locations. Through iterative generative dynamics, they can produce richer posterior samples and better represent complex uncertainty without requiring context-target splits during training. Nevertheless, current approaches often lack an explicit architectural mechanism for

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

encoding the spatial structure of input sets, making it difficult to consistently capture correlations across irregular and unordered observations. The problem propagates to the generative process, these correlations further evolve along the generative time, making the learning of joint spatial and temporal correlations even more challenging. We provide a detailed review of related works in Appendix B.

Contributions. Motivated by these limitations, we propose *Structured Flow Process* (SFP) and summarize our contributions as follows: (1) centered on irregular input sets, we provide SFP, a novel flow-based generative neural process that models function priors over their finite-dimensional marginals; (2) in SFP, we propose *flow-adaptive multihead attention*, which redesigns attention with flow-time conditioning and a multiscale spatial bias to model evolving correlations of finite marginals over irregular input sets along with the generative flow; (3) we validate that SFP preserves the *permutation invariance* of the induced distributions throughout the generative flow; (4) with diversified empirical results, we provide important findings on important topics of prior-posterior interaction, the efficient utilization of irregular inputs, and efficient computation.

2 Background

Setup and Problem. We consider learning a generative model of functions through their finite-dimensional marginals over input sets with arbitrary locations, sizes, and orderings. For any finite input set $X = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)$, the model defines a joint distribution over the corresponding function values $\boldsymbol{y} = (y_1, \ldots, y_N) \in \mathbb{R}^N$, denoted by $p(\boldsymbol{y} \mid X)$. During training, we assume access to a dataset $\{\mathcal{D}^{(i)} = (X^{(i)}, \boldsymbol{y}^{(i)})\}_{i=1}^M$, where $X^{(i)} \in \mathbb{R}^{N_i \times d}$ contains N_i randomly sampled input locations and $\boldsymbol{y}^{(i)} = f^{(i)}(X^{(i)}) \in \mathbb{R}^{N_i}$ denotes the corresponding evaluations of the sampled function $f^{(i)}$. At test time, the model supports both prior and posterior sampling: (1) For any input set X, the model generates prior samples from $p(\boldsymbol{y} \mid X)$ and (2) Given a context set $\mathcal{D}^c = (X^c, \boldsymbol{y}^c)$, it can generate posterior predictive samples at target inputs X^t from $p(\boldsymbol{y}^t \mid X^t, \mathcal{D}^c)$.

2.1 Stochastic Process Priors

The problem defined above can be viewed as learning a stochastic-process (SP) prior, which is defined as a collection of random variables $\{f(\boldsymbol{x})\}_{\boldsymbol{x} \in \mathcal{X}}$. Such a process is characterized by its finite-dimensional marginals $p(f_1, \ldots, f_n)$ with $f_i = f(\boldsymbol{x}_i)$ subject to Kolmogorov consistency conditions (Tao, 2011). These state that all finite-dimensional marginal distributions p are consistent with each other under permutation and marginalization. Let $n \in \mathbb{N}$ and permutation $\alpha \in S_n$; then the following holds for the joint distribution: (1) $p(f_1, \ldots, f_n) = p(f_{\alpha(1)}, \ldots, f_{\alpha(n)})$, and (2) $p(f_i) = \int p(f_1, \ldots, f_n) \prod_{j \neq i} \mathrm{d}f_j$. These conditions further clarify our learning objective: from different marginals collected in the dataset $\mathcal{D}$, we aim to learn a probabilistic model whose finite marginals are mutually compatible across permutations and subsets of inputs.

2.2 Flow Matching

Flow-based generative models (Lipman et al., 2023; Liu et al., 2023) transport a base distribution p_0 to the data distribution p_1 via a velocity field $v : \mathbb{R}^d \times [0, 1] \to \mathbb{R}^d$, governed by the ODE:

$$\mathrm{d}\boldsymbol{y}_t = v_t(\boldsymbol{y}_t)\,\mathrm{d}t, \qquad \boldsymbol{y}_0 \sim p_0. \tag{1}$$

Sampling amounts to integrating equation 1 from $t = 0$ to 1. Flow Matching (FM) designs a probability path $(p_t)_{t \in [0,1]}$ that interpolates between source p_0 and target p_1 and trains a neural network $v_t^\theta(\cdot)$ to learn the velocity field v_t that generates p_t by minimizing the regression loss $\mathbb{E}_{t, \boldsymbol{y}_t} \| v_\theta(\boldsymbol{y}_t, t) - v_t(\boldsymbol{y}_t) \|^2$. Integrating equation 1 with v_t^θ yields an approximate probability flow that transports p_0 to p_1, which is used for generating new samples. While v_t is typically intractable, Lipman et al. (2023) proposes a tractable equivalence called conditional FM (CFM), obtained by conditioning on the target sample $\boldsymbol{y}_1 \sim p_1$: $\mathcal{L}(\theta) = \mathbb{E}_{t, \boldsymbol{y}_1, \boldsymbol{y}_t}\big[\| v_\theta(\boldsymbol{y}_t, t) - v_t(\boldsymbol{y}_t \mid \boldsymbol{y}_1) \|^2\big]$, with $\boldsymbol{y}_0 \sim p_0$ and $\boldsymbol{y}_t \sim p_t(\cdot \mid \boldsymbol{y}_1)$.

2.3 Attention Mechanism

Attention (Vaswani et al., 2017) is permutation-equivariant for modeling dependencies within a set of tokens. Given token representations $\{\boldsymbol{h}_i\}_{i=1}^N$, with $\boldsymbol{h}_i \in \mathbb{R}^D$, self-attention projects each token

into a pair of query, key, and value using learnable matrices $\{W_Q, W_K, W_V\}$: $\boldsymbol{q}_i = W_Q \boldsymbol{h}_i$, $\boldsymbol{k}_i = W_K \boldsymbol{h}_i$, $\boldsymbol{v}_i = W_V \boldsymbol{h}_i$, and computes the output for token i as

$$\tilde{\boldsymbol{h}}_i = \sum_{j=1}^{N} A_{ij} \boldsymbol{v}_j, \qquad A_{ij} = \textsc{SoftMax}\left(\langle \boldsymbol{q}_i, \boldsymbol{k}_j \rangle / \sqrt{D}\right). \tag{2}$$

By computing interactions between all token pairs, self-attention can capture long-range dependencies while remaining agnostic to the ordering of the input set.

Attention Bias The bias $\mathbf{B}$ that incorporates domain knowledge into attention takes the following form: $\tilde{A}_{ij} = \textsc{SoftMax}\left(\langle \boldsymbol{q}_i, \boldsymbol{k}_j \rangle / \sqrt{D} + B_{ij}\right)$, where B_{ij} encodes prior structural information about the interaction between tokens i and j. This can represent relative position (Shaw et al., 2018; Press et al., 2022), graph structure (Lin et al., 2021), or hierarchical relations (Cao & Wang, 2022).

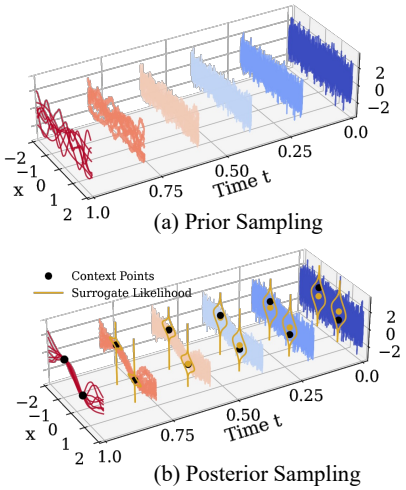
3 Method

In this section, we first present the overall framework of *Structured Flow Processes* (SFP), including parameterization, training, and efficient posterior sampling in Section 3.1. We then describe the proposed Flow-adaptive Multihead Attention (FaMA), designed for random input sets to incorporate their spatial correlations under the time-varying dependency structure induced by the flow process in Section 3.2. Finally, we discuss the properties of SFP in terms of permutation invariance and marginal consistency in Section 3.3.

3.1 Structured Flow Process

SFP constructs a flow-based generative model over functions by modeling their finite-dimensional marginals. Given an input set X, the target distribution is the marginal distribution of the function values $\boldsymbol{y} \sim p(\boldsymbol{y} \mid X)$. For brevity, we write $\boldsymbol{y}|X$ as $\boldsymbol{y}$.

Let $\boldsymbol{y}_1 \coloneqq \boldsymbol{y} \in \mathbb{R}^N$ be a target sample and $\boldsymbol{y}_0 \sim p_0$ be a source sample. We use the conditional OT path and train a neural velocity field $v_\theta(\boldsymbol{y}_t, X, t)$ with the CFM objective $\mathcal{L}(\theta) = \mathbb{E}_{t,(X,\boldsymbol{y}_1),\boldsymbol{y}_0}\left[\|v_\theta(\boldsymbol{y}_t, X, t) - (\boldsymbol{y}_1 - \boldsymbol{y}_0)\|^2\right]$. Once trained, prior samples on any finite input set X can be generated from $p(\boldsymbol{y}|X)$ by solving the flow ODE in Eq. equation 1. Many function space FM approaches define the p_0 through the finite marginals of a GP with a sufficiently regular kernel, which induces a well-defined probability path over function space (Kerrigan et al., 2024; Shi et al., 2025). While theoretically rigorous, such GPs can be costly to sample from and often require task-specific kernels (Pidstrigach et al., 2024; Kollovieh et al., 2025). In practice, many implementations instead use standard Gaussian noise. This is a practical construction and is not covered by existing function-space flow matching theories. Since SFP is defined through finite-dimensional marginals, we follow this practical convention and use $\boldsymbol{y}_0 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, aligned with prior works (Dutordoir et al., 2023; Mathieu et al., 2023; Zhang & Wonka, 2024).



(a) Prior Sampling

(b) Posterior Sampling

Figure 1: Schematic illustration of the SFP sampling scheme.

Posterior Sampling via Context Guided Flow. To compose SFP, there are several methods to perform posterior sampling in flow models. Given context data $\mathcal{D}^c = (X^c, \boldsymbol{y}^c)$ and query locations X^*, we propose efficient sampling from the posterior distribution $p(\boldsymbol{y}^*|X^*, \mathcal{D}^c)$ by solving the flow ODE with the context-guided velocity $v_\theta(\boldsymbol{y}_t|\boldsymbol{y}^c)$ (Song et al., 2021; Ho & Salimans, 2022; Dhariwal & Nichol, 2021; Pokle et al., 2024).

We first construct the joint inputs $X = [X^c, X^*]$ and initialize $\boldsymbol{y}_0 = [\boldsymbol{y}_0^c, \boldsymbol{y}_0^*] \sim p_0$ from the source distribution. The velocity model $v_\theta(\boldsymbol{y}_t, t, X)$ acts on this joint state $\boldsymbol{y}_t = [\boldsymbol{y}_t^c, \boldsymbol{y}_t^*]$. The guided

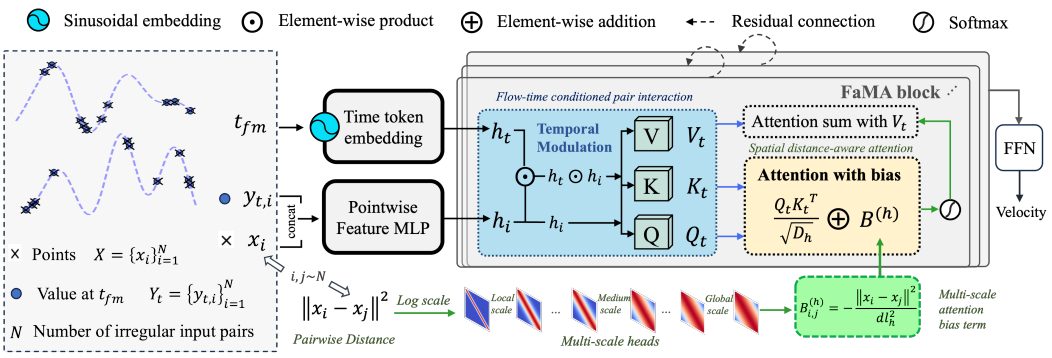


Figure 2: SFP architecture with Flow-adaptive Multihead Attention (FaMA). (Left) SFP takes random function observations and flow-time inputs to learn the velocity field. (Right) Illustration of the FaMA block, which integrates temporal modulation with multiscale spatial bias.

velocity field $v_\theta(\boldsymbol{y}_t|\boldsymbol{y}^c)$ can be decomposed via Bayes' rule as

$$v_\theta(\boldsymbol{y}_t|\boldsymbol{y}^c) = v_\theta(\boldsymbol{y}_t, t, X) + \nabla_{\boldsymbol{y}_t} \log p_t(\boldsymbol{y}^c|\boldsymbol{y}_t),$$

where the first term is the prior from the learned velocity model v_θ, and the second term is the time-dependent likelihood $p_t(\boldsymbol{y}^c|\boldsymbol{y}_t) = \int p(\boldsymbol{y}^c|\boldsymbol{y}_1)p(\boldsymbol{y}_1|\boldsymbol{y}_t)\mathrm{d}\boldsymbol{y}_1$. Directly computing the likelihood $p_t(\boldsymbol{y}^c|\boldsymbol{y}_t)$ is generally intractable, as it requires marginalizing over the intractable $p(\boldsymbol{y}_1|\boldsymbol{y}_t)$. Following prior works on inverse problems (Chung et al., 2023; Daras et al., 2024; Pokle et al., 2024), Tweedie's formula provides an estimate of the conditional expectation: $\hat{\boldsymbol{y}}_{1|t} = \mathbb{E}[\boldsymbol{y}_1 \mid \boldsymbol{y}_t] \approx \boldsymbol{y}_t + (1-t)\, v_\theta(\boldsymbol{y}_t, t, \boldsymbol{x})$. We then introduce a Gaussian surrogate $\tilde{p}(\boldsymbol{y}_1 \mid \boldsymbol{y}_t) := \mathcal{N}(\hat{\boldsymbol{y}}_{1|t}, r_t^2 \boldsymbol{I})$, where $r_t^2 = \frac{(1-t)^2}{(1-t)^2+t^2}$ is the uncertainty derived from the linear interpolation path. For regression tasks, $\boldsymbol{y}^c = P_c \boldsymbol{y}_1 + \boldsymbol{\epsilon}$ where $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma_y^2 \boldsymbol{I})$ and P_c is the projection matrix selecting the context indices. Marginalizing over $\boldsymbol{y}_1$ under the Gaussian surrogate yields the approximate likelihood $p_t(\boldsymbol{y}^c \mid \boldsymbol{y}_t) \approx \int p(\boldsymbol{y}^c \mid \boldsymbol{y}_1)\, \tilde{p}(\boldsymbol{y}_1 \mid \boldsymbol{y}_t)\, d\boldsymbol{y}_1 = \mathcal{N}\big(\hat{\boldsymbol{y}}^c_{1|t}, (r_t^2 + \sigma_y^2)\boldsymbol{I}\big)$, where $\hat{\boldsymbol{y}}^c_{1|t} = P_c \hat{\boldsymbol{y}}_{1|t}$. Taking the gradient with respect to $\boldsymbol{y}_t$ gives the guidance term $\nabla_{\boldsymbol{y}_t} \log p_t(\boldsymbol{y}^c \mid \boldsymbol{y}_t) \approx \nabla_{\boldsymbol{y}_t}(\hat{\boldsymbol{y}}^c_{1|t})^\top \frac{\boldsymbol{y}^c - \hat{\boldsymbol{y}}^c_{1|t}}{r_t^2 + \sigma_y^2}$. We provide algorithms for SFP training and sampling in Appendix D.1, Algorithms 1 and 2.

3.2 Flow-adaptive Multihead Attention

In choosing the model backbone of SFP, we propose a module FaMA in Figure 2 with two key designs. First, *temporal modulation of attention* conditions attention scores on the flow time, enabling the model to capture dependencies that evolve along the probability path. Second, a *multiscale spatial bias* encodes relative spatial structure at multiple scales while remaining compatible with irregular input sets. We elaborate the design as follows.

Temporal Modulation. To model correlations that evolve along the flow, we incorporate self-attention with flow-time conditioning. Rather than using flow time only for global conditioning (Hamad & Rosenbaum, 2025), we use it to modulate the pairwise attention interactions. Specifically, the attention module receives two tokens: a feature token $\boldsymbol{h}_i = \mathrm{MLP}([\boldsymbol{x}_i, y_{t,i}]) \in \mathbb{R}^D$ and a time embedding token $\boldsymbol{h}_{\mathrm{time}} = \mathrm{MLP}(\mathrm{Emb}(t)) \in \mathbb{R}^D$, where Emb is a sinusoidal positional embedding of flow time t. Inspired by gated multiplicative conditioning (Perez et al., 2018; Miyato et al., 2024), we incorporate flow-time conditioning into the attention interactions by defining time-dependent queries, keys, and values as

$$\boldsymbol{q}_i(t) = W_q \boldsymbol{h}_i + U_q(\boldsymbol{h}_i \odot \boldsymbol{h}_{\mathrm{time}}),\ \boldsymbol{k}_i(t) = W_k \boldsymbol{h}_i + U_k(\boldsymbol{h}_i \odot \boldsymbol{h}_{\mathrm{time}}),\ \boldsymbol{v}_i(t) = W_v \boldsymbol{h}_i + U_v(\boldsymbol{h}_i \odot \boldsymbol{h}_{\mathrm{time}}),$$

This design makes attention adapt to flow time t, changing relative attention logits across tokens and enabling the model to capture the evolving dependency structure along the flow path.

Spatial Bias. Correlations among the finite marginals are often closely related to the relative distance between input locations (Wendland, 2004; Bishop & Nasrabadi, 2006). We therefore

incorporate a distance-based bias into the attention logits, allowing the model to explicitly modulate interactions according to the spatial structure of the input set:

$$\tilde{A}^{(h)} = \text{Softmax}\left(\frac{\mathbf{Q}_t^{(h)}\mathbf{K}_t^{(h)\top}}{\sqrt{D_h}} + \mathbf{B}^{(h)}\right), \quad \text{where } B_{ij}^{(h)} = -\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{d\,\ell_h^2}. \tag{3}$$

Here, $\ell_h > 0$ denotes the spatial scale assigned to head h, and the factor d normalizes the squared distance by the input dimension. Although the inputs are normalized to a fixed domain, in our setting, the sampling density and the appropriate spatial scale can vary substantially across different functions. To address the issue, we introduce log-spaced spatial scales across heads, enabling different heads to capture spatial dependencies across different spatial scales. In particular, we parameterize $\ell_h = \exp(s_h c)$, where $s_h = \frac{2(h-1)}{H-1} - 1, h = 1, \ldots, H$, and $c > 0$ control the range of spatial scales. This construction places the H length scales uniformly in log-space over the interval $\log \ell_h \in [-c, c]$. And, this is equivalent to $\ell_h \in [\exp(-c), \exp(c)]$.

Efficient Implementation. When the input sets are large, as in 2D spatial modeling, the efficient while accurate implementation of the attention mechanism becomes essential. Recent advances, such as Rabe & Staats (2021) Dao et al. (2022) and Dao (2024), significantly reduce memory and computational costs. However, incorporating arbitrary bias functions is not naturally supported in these methods. We manage to address the gap with a design in Figure 3.

In our design, the multiscale spatial bias is separable, enabling an efficient attention implementation by avoiding materializing the full bias matrix. The biased attention logits of head h can be written as

$$\begin{aligned} &\langle \boldsymbol{q}_i^{(h)}(t), \boldsymbol{k}_j^{(h)}(t)\rangle / \sqrt{D_h} - \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 / d\ell_h^2 \\ &= -a_i + \langle \bar{\boldsymbol{q}}_i^{(h)}(t), \bar{\boldsymbol{k}}_j^{(h)}(t)\rangle, \end{aligned} \tag{4}$$

where $\bar{\boldsymbol{q}}_i^{(h)}(t) = \left[\boldsymbol{q}_i(t)/D_h^{1/4}; \sqrt{2/d\ell_h^2}\,\boldsymbol{x}_i; 1\right]$, $\bar{\boldsymbol{k}}_j^{(h)}(t) = \left[\boldsymbol{k}_j(t)/D_h^{1/4}; \sqrt{2/d\ell_h^2}\,\boldsymbol{x}_j; -a_j\right]$, and $a_i = \|\boldsymbol{x}_i\|^2/d\ell_h^2$. Since attention is normalized row-wise, the term $-a_i$ is constant across all j for a fixed query i and therefore cancels under the SOFTMAX. As a result, only the key-side correction $-a_j$ needs to be retained, as the attention logits $L_{ij} = \langle \bar{\boldsymbol{q}}_i^{(h)}(t), \bar{\boldsymbol{k}}_j^{(h)}(t)\rangle$ avoid materializing the full pairwise bias matrix compatible with modern acceleration approaches. In our experiments, we adopt FLASHATTENTION (Dao et al., 2022) to improve efficiency on large input sets, with implementation details and efficiency results reported in Appendix D.3.

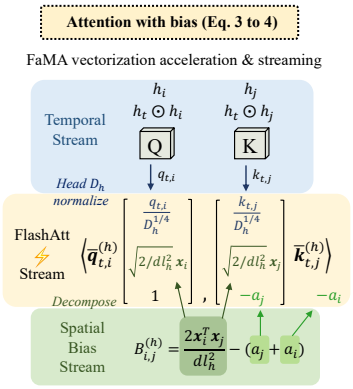


Figure 3: Multiscale biased Attention acceleration with FlashAttention.

3.3 PERMUTATION INVARIANCE AND CONSISTENCY

In this section, we investigate two theoretical properties: permutation invariance and marginal consistency of the learned distributions.

Permutation Invariance. First, we show that the velocity model with FaMA is permutation equivariant in Lemma 3.1. Building on this equivariance, we show that the resulting distribution remains permutation invariant throughout the flow process in Proposition 3.2.

Lemma 3.1. *Let $\boldsymbol{y}_t \in \mathbb{R}^N$ be the flow state at time t paired with input locations $X \in \mathbb{R}^{N\times d}$. Let v_θ be the velocity model parameterized as a transformer model with FaMA block in Section 3.2. Then, for any permutation $\alpha \in S_N$ with the permutation matrix Π_α and any time $t \in [0,1]$, we have:*

$$v_\theta(\Pi_\alpha \boldsymbol{y}_t, \Pi_\alpha \boldsymbol{x}, t) = \Pi_\alpha v_\theta(\boldsymbol{y}_t, \boldsymbol{x}, t).$$

Proposition 3.2 (Permutation Invariance)**.** *Under the Assumption C.1 on v_θ, consider the probability flow governed by the ODE $d\boldsymbol{y}_t/dt = v_\theta(\boldsymbol{y}_t, \boldsymbol{x}, t)$, starting from a base distribution $p_0(\boldsymbol{y}_0|X)$. If the vector field v_θ is equivariant, then $p_t(\boldsymbol{y}_t)$ remains permutation invariant for all $t \in [0, 1-\delta]$ with endpoint truncation $\delta > 0$. That is, for any permutation $\alpha \in S_N$:*

$$p_t(\Pi_\alpha \boldsymbol{y}_t | \Pi_\alpha X) = p_t(\boldsymbol{y}_t|X).$$

We provide detailed proofs for Lemma 3.1 and Proposition 3.2 in Appendix C.1. While Lemma 3.1 focuses on our FaMA block, Proposition 3.2 applies to a broader flow-based generative model (Chen et al., 2018; Song et al., 2021; Lipman et al., 2023). As such, it may also provide support for a wider range of expressive equivariant velocity models used in flow matching for NPs.

Marginal Consistency. Marginal consistency requires that if a portion of the set is marginalized, the marginal distribution will still align with the distribution defined on the original set, i.e., $p(\boldsymbol{y}_{1:k}) = \int p(\boldsymbol{y}_{1:k}, \boldsymbol{y}_{k+1:N}) \mathrm{d}\boldsymbol{y}_{k+1:N}$. We note that, under the conditional OT path, flow matching constructs a marginally consistent probability path as the learning target. When the exact velocity field is available, the flow-induced generative model preserves this property across all finite-dimensional marginals. In practice, this exact field is generally unattainable, and SFP can only approximate the marginal consistency. As shown in Proposition 3.2, the induced distributions of SFP are permutation invariant, which yields a weak form of consistency that for any partition of the data $\boldsymbol{y} = \boldsymbol{y}^* \cup \boldsymbol{y}^c$, the conditionals satisfy: $p_t(\boldsymbol{y}_t^* | \boldsymbol{y}^c) = p_t(\Pi_\alpha \boldsymbol{y}_t^* | \Pi_\beta \boldsymbol{y}^c)$, implying consistency across all orderings induced by the same joint $p_t(\boldsymbol{y}_t)$. More broadly, enforcing exact consistency imposes constraints on the network architecture, often limiting model expressivity (Garnelo et al., 2018b; Foong et al., 2020; Dutordoir et al., 2023). Relaxing such constraints can be advantageous in practice, enabling richer model choices. Training across different input subsets further makes SFP approximate consistency while retaining the flexibility needed for strong empirical performance.

4 Experiment

In this section, we empirically evaluate SFP and competitive baselines on a wide range of tasks. Our evaluation is organized around three perspectives: *prior modeling*, *predictive posterior inference*, and *downstream applications* of probabilistic models. *Prior modeling* assesses the quality of the learned function prior, while *predictive posterior inference* and *downstream evaluation* are widely used settings for benchmarking NP models. We first give the experiment setup in Section 4.1, then present main results with key findings in Section 4.2, and in-depth ablation studies in Section 4.3. Our code and implementation are available at `https://anonymous.4open.science/r/Structured-Flow-Process-62FF`.

4.1 Experiment Settings

Tasks. For prior and posterior evaluation, we use three tasks: mixture-GP emulation, Navier-Stokes (NS) vorticity fields, and global 850 hPa wind fields with zonal and meridional components. These tasks cover synthetic stochastic processes, physical dynamical systems, and real geophysical data, where irregular observations arise naturally. Afterwards, we evaluate SFP as a probabilistic model for downstream tasks through UCI tabular regression and Bayesian optimization. Further details on the training data and task-specific settings are provided in Appendix E and F.

Baselines. Following Table 1, we consider two model families. (1) Conditional NPs, including Attentive Conditional Neural Process (AttCNP) (Kim et al., 2018), Transformer Neural Process (TNP) (Nguyen & Grover, 2022), Latent Neural Process (LNP) (Garnelo et al., 2018b), and Convolutional Conditional Neural Process (ConvCNP) (Garnelo et al., 2018a). (2) The second family, including Neural Diffusion Processes (NDP) (Dutordoir et al., 2023) and Stochastic Process Learning with Operator Flow Matching (SPL-OFM) (Shi et al., 2025), explicitly model prior distributions of functions and supports posterior inference. Our method falls within this family. Detailed implementation settings for SFP and the baseline methods are provided in Appendix D.

Table 1: Comparison of probabilistic function modeling methods. Irregularity is assessed in terms of random observation locations (Loc.), variable observation numbers (Num.), and unordered observations (Ord.).

Method	Irregularity			Modeling		Scalability.
	Loc.	Num.	Ord.	Prior	Post.	
Cond. NPs	✓	✓	✓	✗	✓	✓
SPL-OFM	✗	✓	✗	✓	✓	✗
NDP	✓	✓	✓	✓	✓	✗
SFP	✓	✓	✓	✓	✓	✓

Training and Evaluation. During training, SFP and baselines supporting unordered observations use randomly sampled locations or shuffled input orders, whereas SPL-OFM is trained on regular-grid

Table 2: Prior and posterior performance across different benchmark tasks. Prior quality is evaluated using MMD and SWD. Posterior inference is evaluated using NRMSE and NCRPS with standard deviation across test cases for a fixed training seed. Training-seed variability is evaluated separately in Appendix F.9. Lower values indicate better performance, with the best results shown in bold. "-" indicates cases where training is numerically unstable.

	Prior				Posterior							
	Metric	SPL-OFM	NDP	SFP	Metric	AttCNP	ConvCNP	LNP	TNP	SPL-OFM	NDP	SFP
Mix GPs	MMD	**0.03**	0.05	**0.03**	NRMSE	0.14 ± 0.07	0.35 ± 0.16	0.40 ± 0.18	0.12 ± 0.09	0.06 ± 0.04	0.10 ± 0.07	$\mathbf{0.05 \pm 0.04}$
	SWD	0.08	0.12	**0.06**	NCRPS	0.10 ± 0.03	0.20 ± 0.08	0.23 ± 0.10	0.07 ± 0.04	$\mathbf{0.03 \pm 0.02}$	0.05 ± 0.03	$\mathbf{0.03 \pm 0.02}$
NS System	MMD	0.16	-	**0.14**	NRMSE	0.15 ± 0.03	0.41 ± 0.03	0.49 ± 0.05	0.16 ± 0.02	$\mathbf{0.08 \pm 0.02}$	-	0.10 ± 0.01
	SWD	1.68	-	**1.31**	NCRPS	0.11 ± 0.01	0.23 ± 0.02	0.52 ± 0.03	0.12 ± 0.02	0.06 ± 0.01	-	$\mathbf{0.04 \pm 0.01}$
Wind Field	MMD	0.23	-	**0.22**	NRMSE	0.54 ± 0.03	0.77 ± 0.03	0.80 ± 0.03	0.42 ± 0.03	0.31 ± 0.01	-	$\mathbf{0.28 \pm 0.01}$
	SWD	2.11	-	**1.95**	NCRPS	0.27 ± 0.01	0.39 ± 0.02	0.44 ± 0.01	0.25 ± 0.01	0.22 ± 0.01	-	$\mathbf{0.17 \pm 0.01}$

inputs due to its structural requirement for evenly sampled data. Detailed task-specific settings are provided in Appendix D.2. To evaluate prior modeling performance, we measure the distributional discrepancy between generated and ground-truth samples using Maximum Mean Discrepancy (MMD) and Sliced Wasserstein Distance (SWD). MMD compares samples in a kernel feature space, while SWD compares distributions through random projections. For posterior inference, we report normalized root mean squared error (NRMSE) for predictive accuracy and normalized continuous ranked probability score (NCRPS) for probabilistic evaluation[1]. Both metrics are normalized by the standard deviation of the evaluation dataset. Computation details for all metrics are provided in Appendix F.3.

4.2 Results Analysis

In this section, we provide 4 findings on important topics of prior-posterior interaction, the efficient utilization of irregular inputs, efficient computation, and sparsity handling.

(1) *SFP improves prior modeling while retaining compatibility with training on irregular input sets.*

As shown in Table 2, SFP achieves consistently strong prior modeling performance in terms of MMD and SWD, generally outperforming NDP and improving upon SPL-OFM. The improved performance can be attributed to the proposed FaMA module, which introduces explicit positional structure into the velocity model and facilitates correlation learning over irregular input sets. Specifically, the multiscale spatial bias allows different attention heads to capture correlations over the spatial range, which is important across all evaluated tasks. In mixture-GP emulation, function samples can exhibit varying length scales and smoothness. In the 2D physical tasks, NS vorticity fields contain both localized vortex structures and broader flow patterns, while global wind fields involve regional and large-scale atmospheric correlations. These properties are naturally present in real-world applications and require modeling correlations across multiple spatial scales.

(2) *Process-prior learning improves posterior inference over direct conditional modeling.*

As shown in Table 2, the continuous generative models, SPL-OFM and SFP, generally outperform conditional NP families on both posterior metrics. This suggests that learning a stochastic process prior can provide an effective foundation not only for unconditional generation but also for context-conditioned posterior inference. One reason could be conditional NP models are typically trained with random context-target splits, so each training instance uses only part of a sampled function as conditioning information, which reduces data efficiency for complex spatial functions. In addition, their one-step generation mechanisms may limit their ability to capture rich posterior dependencies over irregular input sets.

Figure 4 provides qualitative support for these results. In the GP task, SFP generates posterior samples that not only fitting the observed context points but also preserve diverse samples in unobserved regions. In contrast, conditional NP baselines tend to produce over-smooth means or limited posterior variability, while NDP and SPL-OFM capture more diversity but are less aligned with the target function. This suggests that SFP better represents the posterior distribution rather than only fitting the conditional mean.

(3) *SFP provides reliable predictive uncertainty for downstream applications.*

[1]We use NCRPS instead of the log likelihood commonly reported in NP studies to measure models that produce non-Gaussian distributions, and NCRPS is a distribution-agnostic proper scoring rule.

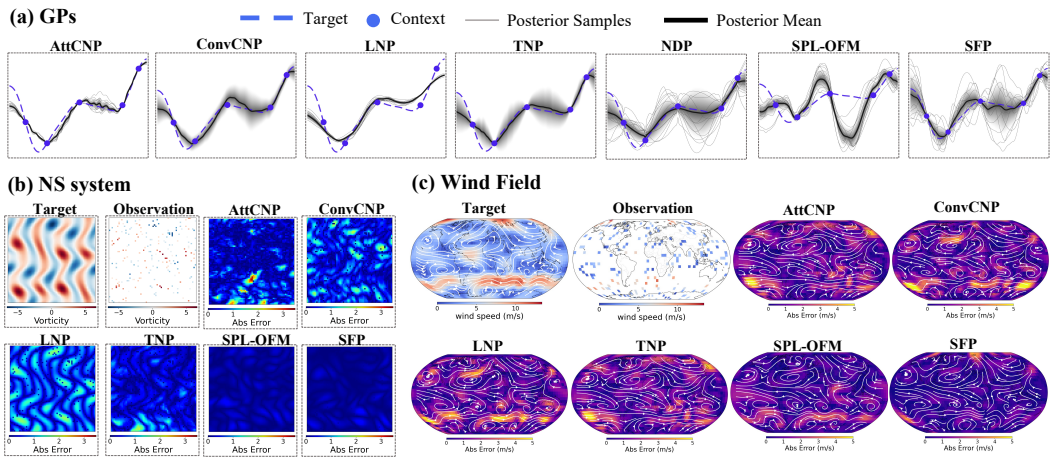


Figure 4: Qualitative results of posterior inference across three tasks: (a) mixture of GPs, (b) Navier–Stokes system, and (c) global wind field. Each panel visualizes the target field and context observations, together with posterior samples for (a) and absolute error maps of the posterior mean predictions for (b) and (c).

SFP also serves as a strong probabilistic model for downstream applications. According to Table 3, SFP achieves low NRMSE across four UCI datasets while consistently obtaining the lowest NCRPS, suggesting that the learned generative flow improves predictive calibration over classical NP baselines. In Bayesian optimization, Figure 5 shows that SFP achieves the best or is competitive across all four benchmark functions. This suggests that SFP provides more informative posterior samples for Thompson sampling, enabling more efficient exploration. Compared with NP baselines, which often reduce regret more slowly or converge to higher regrets, SFP shows stronger performance as a probabilistic surrogate for sequential decision-making.

Table 3: Performance comparison on airfoil self-noise (ASN), concrete compressive strength (CCS), AutoMPG, and Energy Efficiency (EE) from UCI tabular datasets (mean $\pm$ std over 10 random context-query splits).

Task	Metric	AttCNP	ConvCNP	LNP	TNP	SFP
ASN	NRMSE↓	0.58 ± 0.02	0.82 ± 0.03	0.91 ± 0.04	$\mathbf{0.48 \pm 0.01}$	$\mathbf{0.48 \pm 0.02}$
	NCRPS↓	0.27 ± 0.01	0.43 ± 0.02	0.48 ± 0.02	0.24 ± 0.01	$\mathbf{0.21 \pm 0.01}$
CCS	NRMSE↓	0.62 ± 0.03	0.90 ± 0.05	0.96 ± 0.05	0.55 ± 0.03	$\mathbf{0.51 \pm 0.03}$
	NCRPS↓	0.34 ± 0.02	0.54 ± 0.03	0.53 ± 0.02	0.30 ± 0.02	$\mathbf{0.26 \pm 0.02}$
AutoMPG	NRMSE↓	0.52 ± 0.02	0.71 ± 0.02	0.76 ± 0.03	0.41 ± 0.01	$\mathbf{0.38 \pm 0.02}$
	NCRPS↓	0.24 ± 0.01	0.38 ± 0.02	0.41 ± 0.01	0.20 ± 0.01	$\mathbf{0.17 \pm 0.02}$
EE	NRMSE↓	0.55 ± 0.02	0.76 ± 0.03	0.80 ± 0.03	0.45 ± 0.02	$\mathbf{0.42 \pm 0.03}$
	NCRPS↓	0.27 ± 0.01	0.38 ± 0.02	0.42 ± 0.02	0.24 ± 0.01	$\mathbf{0.23 \pm 0.01}$

(4) *The FaMA module enables SFP to learn function distributions in higher dimensional space.*

For the downstream tasks, we construct training sets from high-dimensional GPs with input dimensions up to 8. Unlike NDP and SPL-OFM, SFP remains practically applicable as input dimensionality increases. Empirically, we find that NDP struggles to generalize across higher-dimensional input space, which aligns with prior evidence observed in (Lee et al., 2025), while SPL-OFM requires evenly distributed samples along each dimension, leading to prohibitively large set sizes in higher-dimensional space. For a fixed set size, higher-dimensional input sets become sparser, making it harder to infer spatial correlations from data alone. Multiscale spatial bias addresses this issue by providing structural guidance for learning correlations over sparse input sets. We further investigate this finding in ablation studies on varying sizes of the input set.

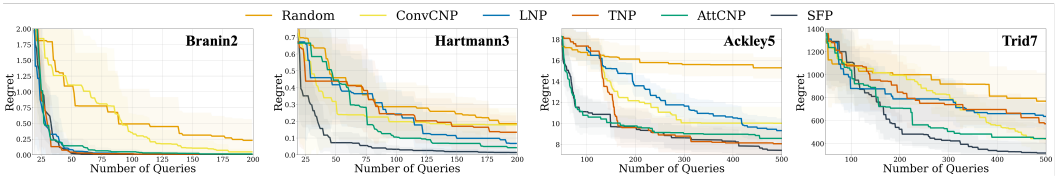


Figure 5: BO performance on benchmark functions, measured by regret as a function of the number of queries. Results are reported as the mean and standard deviation over 20 independent runs with different random seeds.

4.3 Ablation Studies

We revisit the 1D MixGPs and 2D NS systems under different training input-set sizes: $N \in \{16, 32, 64, 128, 256\}$ for the 1D Mix GPs, and $N \in \{16^2, 32^2, 64^2\}$ for the 2D NS system in ablation studies on SFP, using the same evaluation protocol as Section 4.1. Firstly, we evaluate the effectiveness of temporal modulation and multiscale spatial attention bias as two key modules of FaMA. We compare three velocity-model versions as follows: (1) the basic version, where the flow time is appended to each point-wise token, following Hamad & Rosenbaum (2025): $\boldsymbol{h}_i = \mathrm{MLP}([\boldsymbol{y}_{t,i}, \gamma(t), \gamma(\boldsymbol{x}_i)])$, where $\gamma(\cdot)$ denotes the sinusoidal positional encoding, (2) a temporally modulated version, which introduces time-dependent feature modulation into the velocity model; and (3) a further multiscale version, which incorporates the proposed spatial multiscale attention bias. Figure 6 reports the prior modeling and posterior prediction in MMD and NCRPS for each configuration under different input-set sizes.

Temporal modulation and multiscale spatial attention bias consistently improve the performance of SFP in both prior and posterior perspectives as shown in Figure 6: (1) input-time concatenation in basic version modifies individual tokens, affecting on token-to-token interactions which should be learned implicitly from data. Temporal modulation avoids it by retreating attention features with individual time token embedding, which naturally preserves correlations from data and conveys this information along with the flow-adaptive mechanism. Empirically, it consistently improves both prior modeling and posterior prediction, with even larger gains on the 2D NS system. (2) The further introduction of multiscale spatial attention bias leads to significant add-on improvements, particularly in training with small input sets. This dues to given sparse input sets, spatial correlations are difficult to learn from data alone; the proposed bias provides explicit multiscale structural guidance. As the 2D NS system exhibits more complex spatial structures, the benefit of multiscale spatial bias becomes more evident.

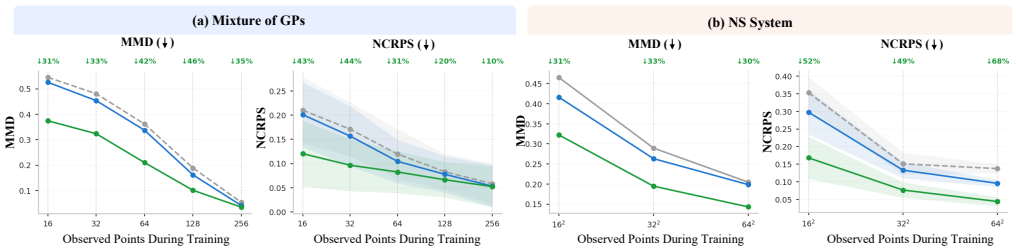


Figure 6: Ablation results on (a) mixture GPs and (b) the NS system under different numbers of observations during training. In each panel, the left plot reports the MMD between generated and true function samples, while the right plot shows the mean and standard deviation of NCRPS on target points conditioned on context points across different cases.

5 Conclusion

In this paper, we introduced *Structured Flow Process* (SFP), a flow-based generative framework for learning function distributions from irregular input sets. SFP models function priors through finite-dimensional marginals. We further proposed Flow-adaptive Multihead Attention (FaMA), which uses flow-time conditioning and multiscale spatial bias to capture evolving correlations, with a FlashAttention-compatible implementation for efficient computation. Our theoretical analysis shows that SFP preserves the permutation invariance of the induced finite-dimensional distributions throughout the generative dynamics. Empirically, SFP achieves strong prior modeling and posterior inference across synthetic, physical, and real-world tasks. The results further show that FaMA improves scalability under sparse and high-dimensional inputs. Downstream tasks further demonstrate that SFP provides reliable predictive uncertainty for probabilistic regression and Bayesian optimization.

Limitations. Although SFP achieves strong performance, its iterative generation makes inference more computationally demanding than single-pass Neural Processes. Additionally, the current implementation is dimension-dependent and requires separate configurations across input dimensions. Developing efficient and dimension-agnostic SFP is an interesting direction for future work.

AI USE STATEMENT

In this work, we used generative AI tools mainly to improve the grammar, clarity, and readability of text originally written by the authors. We did not use generative AI to develop research ideas, formulate claims, design the methodology or experiments, interpret results, or generate figures, data, or experimental results. All AI-assisted edits were reviewed and revised by the authors to ensure that the scientific meaning, claims, citations, and conclusions remained accurate. The authors take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

To support reproducibility, we provide the complete experimental code as part of the supplementary material. The main paper and appendix describe the proposed method, experimental setup, and evaluation protocol, while the supplementary code provides the implementation used to produce the reported results.

REFERENCES

Ayya Alieva, Dmitrii Kochkov, Jamie Alexander Smith, Michael Brenner, Qing Wang, and Stephan Hoyer. Machine learning accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences USA*, 2021.

Matthew Ashman, Cristiana Diaconu, Junhyuck Kim, Lakee Sivaraya, Stratis Markou, James Requeima, Wessel P Bruinsma, and Richard E Turner. Translation equivariant transformer neural processes. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 1924–1944, 2024a.

Matthew Ashman, Cristiana Diaconu, Adrian Weller, Wessel Bruinsma, and Richard Turner. Approximately equivariant neural processes. *Advances in Neural Information Processing Systems*, 37: 97088–97123, 2024b.

Matthew Ashman, Cristiana Diaconu, Eric Langezaal, Adrian Weller, and Richard E Turner. Gridded transformer neural processes for spatio-temporal data. In *Forty-second International Conference on Machine Learning*, 2025.

Sheldon Axler. *Linear algebra done right*. Springer Nature, 2024.

Felix Biggs and Samuel Willis. Llm flow processes for text-conditioned regression, 2026. URL https://arxiv.org/abs/2601.06147.

Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

Wessel Bruinsma, James Requeima, Andrew YK Foong, Jonathan Gordon, and Richard E Turner. The gaussian neural process. In *Third Symposium on Advances in Approximate Bayesian Inference*, 2020.

Wessel Bruinsma, Stratis Markou, James Requeima, Andrew Y. K. Foong, Tom Andersson, Anna Vaughan, Anthony Buonomo, Scott Hosking, and Richard E Turner. Autoregressive conditional neural processes. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=OAsXFPBfTBh.

Shuyang Cao and Lu Wang. Hibrids: Attention with hierarchical biases for structure-aware long document summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 786–807, 2022.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *The Eleventh International Conference on Learning Representations*, 2023.

Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=mZn2Xyh9Ec.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35: 16344–16359, 2022.

Giannis Daras, Hyungjin Chung, Chieh-Hsin Lai, Yuki Mitsufuji, Jong Chul Ye, Peyman Milanfar, Alexandros G Dimakis, and Mauricio Delbracio. A survey on diffusion models for inverse problems. *arXiv preprint arXiv:2410.00083*, 2024.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Hongkun Dou, Junzhe Lu, Zeyu Li, Xiaoqing Zhong, Wen Yao, Lijun Yang, and Yue Deng. Score-based neural processes. *IEEE Transactions on Neural Networks and Learning Systems*, 36(6): 11589–11601, 2025. doi: 10.1109/TNNLS.2025.3527223.

Yann Dubois, Jonathan Gordon, and Andrew YK Foong. Neural process family. http://yanndubs.github.io/Neural-Process-Family/, September 2020.

Vincent Dutordoir, Alan Saul, Zoubin Ghahramani, and Fergus Simpson. Neural diffusion processes. In *International Conference on Machine Learning*, pp. 8990–9012. PMLR, 2023.

Andrew Foong, Wessel Bruinsma, Jonathan Gordon, Yann Dubois, James Requeima, and Richard Turner. Meta-learning stationary stochastic process prediction with convolutional neural processes. *Advances in Neural Information Processing Systems*, 33:8284–8295, 2020.

Giulio Franzese, Giulio Corallo, Simone Rossi, Markus Heinonen, Maurizio Filippone, and Pietro Michiardi. Continuous-time functional diffusion processes. *Advances in Neural Information Processing Systems*, 36:37370–37400, 2023.

Marta Garnelo, Dan Rosenbaum, Christopher Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo Rezende, and SM Ali Eslami. Conditional neural processes. In *International conference on machine learning*, pp. 1704–1713. PMLR, 2018a.

Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J Rezende, SM Eslami, and Yee Whye Teh. Neural processes. *arXiv preprint arXiv:1807.01622*, 2018b.

Aditya Gopalan, Shie Mannor, and Yishay Mansour. Thompson sampling for complex online problems. In *International conference on machine learning*, pp. 100–108. PMLR, 2014.

Jonathan Gordon, Wessel P. Bruinsma, Andrew Y. K. Foong, James Requeima, Yann Dubois, and Richard E. Turner. Convolutional conditional neural processes. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=Skey4eBYPS.

Ling Guo, Hao Wu, and Tao Zhou. Normalizing field flows: Solving forward and inverse stochastic differential equations using physics-informed flow models. *Journal of Computational Physics*, 461:111202, 2022.

Hussen Abu Hamad and Dan Rosenbaum. Flow matching neural processes. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=R8n2h7hzqS.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

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

Peter Holderrieth, Michael J Hutchinson, and Yee Whye Teh. Equivariant learning of stochastic fields: Gaussian processes and steerable conditional neural processes. In *International conference on machine learning*, pp. 4297–4307. PMLR, 2021.

Gavin Kerrigan, Justin Ley, and Padhraic Smyth. Diffusion generative models in infinite dimensions. In *International Conference on Artificial Intelligence and Statistics*, pp. 9538–9563. PMLR, 2023.

Gavin Kerrigan, Giosue Migliorini, and Padhraic Smyth. Functional flow matching. In *International Conference on Artificial Intelligence and Statistics*, pp. 3934–3942. PMLR, 2024.

Hyunjik Kim, Andriy Mnih, Jonathan Schwarz, Marta Garnelo, Ali Eslami, Dan Rosenbaum, Oriol Vinyals, and Yee Whye Teh. Attentive neural processes. In *International Conference on Learning Representations*, 2018.

Dmitrii Kochkov, Jamie A. Smith, Ayya Alieva, Qing Wang, Michael P. Brenner, and Stephan Hoyer. Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, 118(21), 2021. ISSN 0027-8424. doi: 10.1073/pnas.2101784118. URL https://www.pnas.org/content/118/21/e2101784118.

Marcel Kollovieh, Marten Lienen, David Lüdke, Leo Schwinn, and Stephan Günnemann. Flow matching with gaussian process priors for probabilistic time series forecasting. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=uxVBbSlKQ4.

Tuan Anh Le, Hyunjik Kim, Marta Garnelo, Dan Rosenbaum, Jonathan Schwarz, and Yee Whye Teh. Empirical evaluation of neural process objectives. In *NeurIPS workshop on Bayesian Deep Learning*, volume 4, 2018.

Hyungi Lee, Eunggu Yun, Giung Nam, Edwin Fong, and Juho Lee. Martingale posterior neural processes. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=-9PVqZ-IR_.

Hyungi Lee, Chaeyun Jang, Dong Bok Lee, and Juho Lee. Dimension agnostic neural processes. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=uGJxl2odR0.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, pp. 3744–3753. PMLR, 2019.

Juho Lee, Yoonho Lee, Jungtaek Kim, Eunho Yang, Sung Ju Hwang, and Yee Whye Teh. Bootstrapping neural processes. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6606–6615, 2020.

Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=c8P9NQVtmnO.

Zongyi Li, Daniel Zhengyu Huang, Burigede Liu, and Anima Anandkumar. Fourier neural operator with learned deformations for pdes on general geometries. *Journal of Machine Learning Research*, 24(388):1–26, 2023.

Jae Hyun Lim, Nikola B Kovachki, Ricardo Baptista, Christopher Beckham, Kamyar Azizzadenesheli, Jean Kossaifi, Vikram Voleti, Jiaming Song, Karsten Kreis, Jan Kautz, et al. Score-based diffusion models in function space. *Journal of Machine Learning Research*, 26(158):1–62, 2025.

Kevin Lin, Lijuan Wang, and Zicheng Liu. Mesh graphormer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 12939–12948, 2021.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023.

Xingchao Liu, Chengyue Gong, et al. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023.

Christos Louizos, Xiahan Shi, Klamer Schutte, and Max Welling. The functional neural process. *Advances in Neural Information Processing Systems*, 32, 2019.

Emile Mathieu, Vincent Dutordoir, Michael Hutchinson, Valentin De Bortoli, Yee Whye Teh, and Richard Turner. Geometric neural diffusion processes. *Advances in Neural Information Processing Systems*, 36:53475–53507, 2023.

Takeru Miyato, Bernhard Jaeger, Max Welling, and Andreas Geiger. GTA: A geometry-aware attention mechanism for multi-view transformers. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=uJVHygNeSZ.

Tung Nguyen and Aditya Grover. Transformer neural processes: Uncertainty-aware meta learning via sequence modeling. In *International Conference on Machine Learning*, pp. 16569–16594. PMLR, 2022.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Victor Picheny, Joel Berkeley, Henry B Moss, Hrvoje Stojic, Uri Granta, Sebastian W Ober, Artem Artemev, Khurram Ghani, Alexander Goodall, Andrei Paleyes, et al. Trieste: Efficiently exploring the depths of black-box functions with tensorflow. *arXiv preprint arXiv:2302.08436*, 2023.

Jakiw Pidstrigach, Youssef Marzouk, Sebastian Reich, and Sven Wang. Infinite-dimensional diffusion models. *Journal of Machine Learning Research*, 25(414):1–52, 2024.

Ashwini Pokle, Matthew J. Muckley, Ricky T. Q. Chen, and Brian Karrer. Training-free linear image inverses via flows. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=PLIt3a4yTm.

Ofir Press, Noah Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=R8sQPpGCv0.

Markus N Rabe and Charles Staats. Self-attention does not need $o(n^2)$ memory. *arXiv preprint arXiv:2112.05682*, 2021.

Stephan Rasp, Stephan Hoyer, Alexander Merose, Ian Langmore, Peter Battaglia, Tyler Russell, Alvaro Sanchez-Gonzalez, Vivian Yang, Rob Carver, Shreya Agrawal, et al. Weatherbench 2: A benchmark for the next generation of data-driven global weather models. *Journal of Advances in Modeling Earth Systems*, 16(6):e2023MS004019, 2024.

Tommy Rochussen and Vincent Fortuin. Sparse Gaussian neural processes. In James Urquhart Allingham and Siddharth Swaroop (eds.), *Proceedings of the 7th Symposium on Advances in Approximate Bayesian Inference*, volume 289 of *Proceedings of Machine Learning Research*, pp. 194–219. PMLR, 29 Apr 2025. URL https://proceedings.mlr.press/v289/rochussen25a.html.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pp. 464–468, 2018.

Ying Shen, Zhiyang Xu, Jiuhai Chen, Shizhe Diao, Jiaxin Zhang, Yuguang Yao, Joy Rimchala, Ismini Lourentzou, and Lifu Huang. Latte-flow: Layerwise timestep-expert flow-based transformer. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=dxVnCJRSm8.

Yaozhong Shi, Angela F. Gao, Zachary E. Ross, and Kamyar Azizzadenesheli. Universal functional regression with neural operator flows. *Transactions on Machine Learning Research*, 2024.

Yaozhong Shi, Zachary E Ross, Domniki Asimaki, and Kamyar Azizzadenesheli. Stochastic process learning via operator flow matching. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=Quvnn2o17a.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

Terence Tao. *An introduction to measure theory*, volume 126. American Mathematical Soc., 2011.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Qi Wang and Herke Van Hoof. Doubly stochastic variational inference for neural processes with hierarchical latent variables. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 10018–10028. PMLR, 2020.

Sifan Wang, Zehao Dou, Siming Shan, Tong-Rui Liu, and Lu Lu. Fundiff: Diffusion models over function spaces for physics-informed generative modeling. *Nature Communications*, 2026.

Yuyang Wang, Anurag Ranjan, Joshua M. Susskind, and Miguel Ángel Bautista. INRFlow: Flow matching for INRs in ambient space. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=qIzYWidWZH.

Holger Wendland. *Scattered data approximation*, volume 17. Cambridge university press, 2004.

Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

Jin Xu, Emilien Dupont, Kaspar Märtens, Thomas Rainforth, and Yee Whye Teh. Deep stochastic processes via functional markov transition operators. *Advances in Neural Information Processing Systems*, 36:37975–37994, 2023.

Wenzhe Yin, Zehao Xiao, Jiayi Shen, Yunlu Chen, Cees GM Snoek, Jan-Jakob Sonke, and Efstratios Gavves. Geometric neural process fields. *arXiv preprint arXiv:2502.02338*, 2025.

Biao Zhang and Peter Wonka. Functional diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4723–4732, 2024.

A NOTATION

Table 4: Summary of notation.

Symbol	Meaning
X	Input locations (set / grid of query points)
X^c, X^*	Context and query input locations
$\boldsymbol{y}$	Function values / observations at inputs
$\boldsymbol{y}^c$, $\boldsymbol{y}^*$	Context and query outputs
t	Flow / diffusion time
$\boldsymbol{y}_0$	Source noise (base) sample
$\boldsymbol{y}_1$	Data sample (clean endpoint)
$\boldsymbol{y}_t$	Interpolated state at time t
$p_t(\boldsymbol{y}_t \mid X)$	Density of the flow state $\boldsymbol{y}_t$ (conditioned on X)
$v_\theta(\boldsymbol{y}_t, t, X)$	Neural velocity field with parameters θ
$\mathcal{T}$	Training distribution / dataset of functions
N	Number of input-output pairs in a function instance
K	Number of solver steps / function evaluations at sampling time
σ_y^2	Observation noise variance
P_c	Projection / selection operator for context indices

B RELATED WORKS.

Advanced neural processes. Neural processes (NPs) provide an amortized stochastic-process view of conditional prediction, beginning with conditional NPs and latent-variable NPs (Garnelo et al., 2018a;b). Recent advances in NPs have improved their expressiveness through both architectural designs and richer uncertainty mechanisms. Attentive NPs mitigate context underfitting by constructing target-specific context summaries (Kim et al., 2018). Convolutional CNPs and ConvNPs encode translation equivariance and support coherent samples for stationary stochastic processes (Gordon et al., 2020; Foong et al., 2020). Gaussian NPs explicitly model predictive correlations across target locations (Bruinsma et al., 2020). Other architectural designs incorporate equivariance and symmetry-aware inductive biases (Holderrieth et al., 2021; Ashman et al., 2024a;b), geometric bases for neural fields (Yin et al., 2025), and gridded pseudo-token processors for large spatio-temporal data (Ashman et al., 2025).

A complementary line enriches the stochastic mechanism. Functional NPs learn dependency graphs over latent point representations (Louizos et al., 2019), while hierarchical and doubly stochastic variants combine global and local latent variables to capture target-specific variation (Wang & Van Hoof, 2020). Bootstrap NPs and Martingale Posterior NPs provide more data-driven sources of functional uncertainty without imposing a fixed finite-dimensional latent form (Lee et al., 2020; 2023). Markov NPs construct stochastic processes through functional Markov transition operators, emphasizing exchangeability and marginal consistency (Xu et al., 2023). Transformer NPs (TNPs) cast conditional prediction as sequence modeling with autoregressive transformers (Nguyen & Grover, 2022); related autoregressive conditional NPs induce dependent and non-Gaussian predictive distributions at test time (Bruinsma et al., 2023). Dimension Agnostic NPs extend NP-style regression across varying input and output dimensions (Lee et al., 2025). More recent generative extensions, such as Flow Matching NPs, combine NP conditioning with ODE-based continuous generative flows, enabling expressive context-conditioned function prediction and parallel conditional sampling (Hamad & Rosenbaum, 2025). This line of NP work demonstrates that expressive predictive modeling depends jointly on set/sequence architectures, structural inductive biases, and the stochastic mechanism used to represent functional uncertainty.

Continuous generative models on function space. Recent work extends diffusion, score-based modeling, and flow matching to infinite-dimensional function spaces, aiming to model stochastic functions independently of a fixed discretization. Early theoretical frameworks formulate noising and denoising dynamics directly over Hilbert or related function spaces beyond finite-dimensional discretizations (Franzese et al., 2023; Kerrigan et al., 2023; Pidstrigach et al., 2024; Lim et al., 2025). Functional Flow Matching learns vector fields that transport a reference Gaussian measure to a target

distribution over functions (Kerrigan et al., 2024). Operator Flow Matching further learns stochastic-process priors through neural operators and provides densities for arbitrary finite collections of function values (Shi et al., 2025). Related normalizing-flow and operator-flow models map between Gaussian-process priors and data function spaces, supporting likelihood-based functional regression from scattered measurements (Guo et al., 2022; Shi et al., 2024).

In practical implementation, many function-space generative models rely on neural operator backbones such as Fourier neural operators (Li et al., 2021). These architectures are efficient on structured or uniformly discretized domains, but their reliance on Fourier bases, meshes, or learned deformations can make arbitrary off-grid evaluation and irregular-domain conditioning nontrivial; Geo-FNO and related geometric operators partly relax this restriction (Li et al., 2023). Another line learns function distributions through finite-dimensional marginals over arbitrary input sets. Neural Diffusion Processes (NDPs) instantiate this idea by learning stochastic processes through their finite marginals and incorporating exchangeability into the denoising architecture (Dutordoir et al., 2023). Geometric NDPs extend NDPs with symmetry-aware noising processes and equivariant score networks for Euclidean and non-Euclidean fields (Mathieu et al., 2023). More recent variants, including score-based NPs, Neural Bridge Processes, and physics-informed function-space diffusion, further strengthen correlated non-Gaussian sampling, input-coupled diffusion trajectories, and physical constraints (Dou et al., 2025; Wang et al., 2026). We follow the finite-marginal perspective of NDP models because it remains naturally compatible with arbitrary context and target locations while leaving open questions about marginal consistency, scalable conditioning, and efficient generation over large target sets.

Flow-time conditioning and ambient-space flow matching. Recent flow-based architectures also incorporate time conditioning for other purposes. INRFlow (Wang et al., 2025) encodes a coordinate–value set into a shared representation and uses a conditionally factorized point-wise objective. Although its point-wise predictions are coupled through the shared representation, SFP models dependencies through direct token-to-token interactions within every FaMA layer. LaTtE-Flow (Shen et al., 2026) conditions attention on flow time to accelerate image generation by assigning different layer groups to different timestep intervals and reusing cross-layer attention. FaMA serves a different architectural role: it modulates the queries, keys, and values in every layer throughout the flow and combines this modulation with head-specific multiscale distance biases for irregular function observations.

C PROOFS

C.1 PROOF OF LEMMA 3.2 AND THEOREM 3.3

Proof of Lemma 3.1. In general, self-attention without order-dependent positional encodings is permutation equivariant, a property widely used in set-based Transformer architectures (Vaswani et al., 2017; Lee et al., 2019). Here, we particularly analyze our proposed architecture by examining its three main components: the pointwise embedding, the flow-adaptive multihead attention (FaMA) mechanism, and the output projection.

(1) Pointwise embedding: The input tokens $\boldsymbol{H} \in \mathbb{R}^{N\times D}$ are constructed via a pointwise MLP applied to the concatenated inputs: $\boldsymbol{h}_i = \text{MLP}([\boldsymbol{x}_i, y_{t,i}])$. Let the inputs be permuted: $X_\alpha = \Pi_\alpha X$ and $\boldsymbol{y}_{t,_\alpha} = \Pi_\alpha \boldsymbol{y}_t$. The i-th row of the permuted input corresponds to the $\alpha(i)$-th row of the original input. Since the MLP processes each token independently, the resulting feature matrix transforms as:

$$\boldsymbol{H}_\alpha = \Pi_\alpha \boldsymbol{H}.$$

(2) FaMA: For head h, let $\boldsymbol{Q} = \boldsymbol{H}W_Q + (\boldsymbol{H} \odot \boldsymbol{H}_{\text{time}})U_Q, \boldsymbol{K} = \boldsymbol{H}W_K + (\boldsymbol{H} \odot \boldsymbol{H}_{\text{time}})U_K$, and $\boldsymbol{V} = \boldsymbol{H}W_V + (\boldsymbol{H} \odot \boldsymbol{H}_{\text{time}})U_V$. The $\boldsymbol{h}_{\text{time}} = \text{MLP}([\text{Emb}(t)])$ and Emb denotes the sinusoidal time embedding. Since $\boldsymbol{H}_{\text{time}}$ is shared across all tokens as $\boldsymbol{H}_{\text{time}} = \boldsymbol{1}_N \boldsymbol{h}_{\text{time}}^\top$, it is invariant to input permutations. Therefore,

$$(\Pi_\alpha \boldsymbol{H}) \odot \boldsymbol{H}_{\text{time}} = \Pi_\alpha(\boldsymbol{H} \odot \boldsymbol{H}_{\text{time}}).$$

Hence, the temporal modulation of $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$ preserves permutation equivariance:

$$\boldsymbol{Q}_\alpha = \Pi_\alpha \boldsymbol{Q}, \quad \boldsymbol{K}_\alpha = \Pi_\alpha \boldsymbol{K}, \quad \boldsymbol{V}_\alpha = \Pi_\alpha \boldsymbol{V}.$$

The attention logits L consist of two terms:

$$L_{ij} = \frac{\boldsymbol{q}_i^\top \boldsymbol{k}_j}{\sqrt{D_h}} + B_{ij}, \quad \text{where } B_{ij} = -\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{d\, l_h^2}.$$

For the permuted inputs, the term for indices (i, j) corresponds to the original indices $(\alpha(i), \alpha(j))$. In matrix notation, both the inner product matrix $\boldsymbol{Q}\boldsymbol{K}^\top$ and the spatial bias matrix $\boldsymbol{B}$ transform via similarity transformation with Π_α. Thus, we have $L_\alpha = \Pi_\alpha L \Pi_\alpha^\top$. Since the Softmax is applied row-wise, the normalized attention matrix $A = \text{Softmax}(L)$ follows the same transformation:

$$A_\alpha = \Pi_\alpha A \Pi_\alpha^\top.$$

Denote the output of the attention block as $\boldsymbol{O} = A\boldsymbol{V}$. With permuted values $\boldsymbol{V}_\alpha = \Pi_\alpha \boldsymbol{V}$:

$$\boldsymbol{O}_\alpha = A_\alpha \boldsymbol{V}_\alpha = (\Pi_\alpha A \Pi_\alpha^\top)(\Pi_\alpha \boldsymbol{V}) = \Pi_\alpha A (\Pi_\alpha^\top \Pi_\alpha)\boldsymbol{V}.$$

Since Π_α is orthogonal, we have $\boldsymbol{O}_\alpha = \Pi_\alpha(A\boldsymbol{V}) = \Pi_\alpha \boldsymbol{O}$.

(3) Output Projection: The output layer is computed via a pointwise MLP on $\boldsymbol{O}$. Analogous to the *Pointwise Embedding*, this operation preserves permutation equivariance.

Consequently, since v_θ is a composition of permutation equivariant functions, it satisfies:

$$v_\theta(\Pi_\alpha \boldsymbol{y}_t, \Pi_\alpha X, t) = \Pi_\alpha v_\theta(\boldsymbol{y}_t, X, t)$$

which completes the proof. □

Assumption C.1 (Regularity of the velocity model)**.** *For input set X and endpoint truncation $\delta > 0$, assume that $v_\theta(\cdot, X, t)$ is globally Lipschitz and continuously differentiable with respect to $\boldsymbol{y}$, and that $v_\theta(\boldsymbol{y}, X, \cdot)$ is continuous in t, for all $t \in [0, 1-\delta]$.*

Proof of Theorem 3.2. Given assumption C.1, the evolution of the log-density in a flow generation is given by the instantaneous change of variables formula (Chen et al., 2018; Lipman et al., 2023):

$$\log p_t(\boldsymbol{y}_t|X) = \log p_0(\boldsymbol{y}_0|X) - \int_0^t \text{Tr}\left(\frac{\partial v_\theta}{\partial \boldsymbol{y}_\tau}(\boldsymbol{y}_\tau, X, \tau)\right) \mathrm{d}\tau. \tag{5}$$

We prove the $p_t(\boldsymbol{y}_t|X)$ is exchangeable by showing the two terms in the RHS of Equation equation 5.

(1) Initial Term: By assumption, the base distribution is exchangeable $\alpha \in S_N$: $\log p_0(\Pi_\alpha \boldsymbol{y}_0|X) = \log p_0(\boldsymbol{y}_0|X)$.

(2) Integral Term: For the integral term, let $\boldsymbol{y}_\tau$ (integrand) be the trajectory leading to $\boldsymbol{y}_t$. By the equivariance of the velocity model (Lemma 3.1), we have $v_\theta(\Pi_\alpha \boldsymbol{y}_\tau, \Pi_\alpha X, \tau) = \Pi_\alpha v_\theta(\boldsymbol{y}_\tau, X, \tau)$.

Let $J(\boldsymbol{y}_\tau, X) = \frac{\partial v_\theta}{\partial \boldsymbol{y}_\tau}$ denote the Jacobian of the velocity field. Differentiating the equivariance condition $v_\theta(\Pi_\alpha \boldsymbol{y}_\tau, \Pi_\alpha X, \tau) = \Pi_\alpha v_\theta(\boldsymbol{y}_\tau, X, \tau)$ with respect to $\boldsymbol{y}_\tau$ yields the similarity transformation:

$$J(\Pi_\alpha \boldsymbol{y}_\tau, \Pi_\alpha X) = \Pi_\alpha J(\boldsymbol{y}_\tau, X)\Pi_\alpha^\top.$$

By the cyclic property of the trace operator (Axler, 2024, Section 10.A), the integrand remain unchanged:

$$\mathrm{Tr}(J(\Pi_\alpha \boldsymbol{y}_\tau, \Pi_\alpha X)) = \mathrm{Tr}(\Pi_\alpha J(\boldsymbol{y}_\tau, X)\Pi_\alpha^\top) = \mathrm{Tr}(J(\boldsymbol{y}_\tau, X)).$$

Consequently, both the base probability and the accumulated change in density are identical, implying

$$p_t(\Pi_\alpha \boldsymbol{y}_t|\Pi_\alpha X) = p_t(\boldsymbol{y}_t|X),$$

where we complete the proof. □

D Implementation Details

D.1 Pseudo algorithm

Algorithm 1: SFP Training

Require: Training dataset $\mathcal{T}$

Initialize v_θ
while not converged **do**
  Sample $(\boldsymbol{x}, \boldsymbol{y}_1) \sim \mathcal{T}$
  $t \sim \mathcal{U}[0, 1]$
  $\boldsymbol{y}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  $\boldsymbol{y}_t \leftarrow (1-t)\boldsymbol{y}_0 + t\boldsymbol{y}_1$
  $\mathcal{L}(\theta) \leftarrow \|v_\theta(\boldsymbol{y}_t, t, \boldsymbol{x}) - u_t\|^2$
  Update θ with $\nabla_\theta \mathcal{L}$
end while

Algorithm 2: SFP Sampling

Require: Context $(X^c, \boldsymbol{y}^c)$, Query X^*, K, λ

$X \leftarrow [X^c, X^*]$, $\boldsymbol{y}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $\Delta t \leftarrow 1/K$
for $k = 0$ to $K - 1$ **do**
  $t \leftarrow k\Delta t$, $r_t^2 \leftarrow \dfrac{(1-t)^2}{(1-t)^2 + t^2}$
  $\hat{\boldsymbol{y}}_{1|t} \leftarrow \boldsymbol{y}_t + (1-t)v_\theta(\boldsymbol{y}_t, t, \boldsymbol{x})$
  $g \leftarrow \nabla_{\boldsymbol{y}_t}(\hat{\boldsymbol{y}}_{1|t})^\top \Big((\boldsymbol{y}^c - \hat{\boldsymbol{y}}^c_{1|t})/(r_t^2 + \sigma_y^2)\Big)$
  $\rho_t = \lambda / \|\boldsymbol{y}^c - \hat{\boldsymbol{y}}^c_{1|t}\|$
  $v \leftarrow \begin{cases} v_\theta + \rho_t g\,, & \text{if posterior sampling,} \\ v_\theta\,, & \text{if prior sampling} \end{cases}$
  $\boldsymbol{y}_{t+\Delta t} \leftarrow \boldsymbol{y}_t + v\Delta t$
end for

Figure 7: **Left:** Training procedure. **Right:** Posterior sampling via guided flow.

D.2 Baselines

- **AttCNP**: The Attentive Neural Process (Kim et al., 2018) uses an attention mechanism to augment the NPs, which consists of a cross-attention encoder for local context aggregation and a self-attentive pooling encoder for global latent variable inference. We use the implementation provided in the `Neural Process Family`[2] library (Dubois et al., 2020).
- **ConvCNP**: We evaluate Conditional Neural Processes (Garnelo et al., 2018a), which encode context points into a global representation via mean-aggregation without introducing a stochastic latent variable. We use the `Neural Process Family` implementation (Dubois et al., 2020), which encodes context using concatenated pooling layers and decodes the predictive distribution via a multi-layer perceptron.
- **TNP**: We include two variants of the Transformer Neural Processes framework (Nguyen & Grover, 2022). We use the official implementation of TNP at `https://github.com/tung-nd/TNP-pytorch`.
- **NDP**: We adopt the official implementation of the Neural Diffusion Process (NDP) (Dutordoir et al., 2023). A DDPM diffusion formulation (Ho et al., 2020) is used with 1000 diffusion steps for noise scheduling. The denoising network comprises five bi-dimensional attention blocks, with configuration set to the defaults of the original implementation [3]. At test time, a repainting-style inference procedure is applied: noise is added to the context targets $\boldsymbol{y}$, which are then concatenated with the full test set and iteratively denoised to generate predictions.
- **SPL-OFM**: We use the publicly released implementation from `https://github.com/yzshi5/spl_ofm`. SPL-OFM is a stochastic process learning method based on operator flow matching that models function distributions through flow-based generation. It supports both prior generation and posterior inference, making it a strong baseline aligned with our setting. In its current implementation, SPL-OFM typically uses on the order of 10^2 function evaluations for generation, depending on the solver tolerance. The model backbone consists of a time-conditioned Fourier Neural Operator with 4 layers, 16 modes per spatial dimension, hidden dimension 128. We follow the original implementation and set atol $=$

[2] `https://github.com/YannDubs/Neural-Process-Family`
[3] `https://github.com/vdutor/neural-diffusion-processes`

rtol $= 1 \times 10^{-5}$. For posterior sampling, SPL-OFM uses stochastic gradient Langevin dynamics (SGLD) initialized from the maximum a posteriori estimate, collects samples every 10 iterations, and sets the noise temperature to 1. For 1D GP, we use 4×10^4 SGLD iterations with 3×10^3 burn-in iterations, a learning rate decayed from 5×10^{-3} to 4×10^{-3}, 32 Hutchinson samples, and a noise level of 10^{-2}. For the NS system, we follow the same setting as in the original paper: 2×10^4 iterations with 3×10^3 burn-in iterations, a learning rate decayed from 3×10^{-3} to 2×10^{-3}, 8 Hutchinson samples, and a noise level of 10^{-3}.

A key difference in the experimental setting is that **SPL-OFM is trained on regular grids**, whereas the other approaches are trained with irregular observations. In the 1D GP experiments, we randomly sample input locations from the domain, with input-set sizes ranging from 32 to 256. Since SPL-OFM requires evenly spaced observations, we train it on uniformly spaced grids, with the number of points randomly chosen from $(32, 64, 128, 256)$. For the 2D experiments, SPL-OFM is trained on a fixed 64×64 grid, whereas the other models are trained on the same grid with randomly shuffled input order, enabling evaluation under unordered observation sets.

Specifically, in the 1D GP experiments, we train SPL-OFM on uniformly spaced grids with varying set sizes and train the other methods on randomly sampled input locations. In the 2D experiments, we train SPL-OFM on a fixed regular grid, whereas the other methods use the same data with a randomly shuffled input order. Detailed settings are provided in Appendix F.

D.3 TRAINING

This section introduces the detailed settings for each experiment.

All SFP experiments use the same backbone architecture with the FaMA block, as shown in Figure 2. The training protocol, summarized in Table 5. We optimize all models with AdamW and a cosine annealing learning-rate schedule. The learning rate is initialized at 5×10^{-4} and decayed to 1×10^{-6} over 128 epochs for all experiments. We use an exponential moving average (EMA) of model parameters with a decay rate 0.999 for evaluation. Training is parallelized over four NVIDIA A100 40GiB GPUs, with each experimental run assigned to a single GPU.

Table 5: Model and Training configuration.

Config.	**Value**	**Config.**	**Value**
Optimizer	AdamW	LR schedule	Warmup + cosine
LR	5×10^{-4}	Epochs	128
Weight decay	10^{-4}	EMA rate	0.999
Hidden dim.	128	Layers	6
Heads	4	Attn. dropout	0.1
FM loss	MSE		

FlashAttention. We implement FaMA attention using exact FlashAttention. As described in Section 3.2, we rewrite the FaMA score as an augmented query-key dot product, avoiding the explicit materialization of the dense spatial-bias matrix while preserving the exact FaMA formulation. This allows us to use JAX's fused FlashAttention backend with the same model configuration as reported in Table 5. By avoiding explicit $O(N^2)$ attention-map storage, FlashAttention substantially reduces peak memory for tasks with larger input sets and also lowers runtime overhead in our measurements. Table 6 reports the resulting memory and runtime improvements.

Cost of Posterior Guidance. Posterior guidance does not explicitly materialize the Jacobian. The derivative is taken with respect to the flow state $\boldsymbol{y}_t \in \mathbb{R}^{N \times p}$, rather than the input locations $X \in \mathbb{R}^{N \times d}$, and is computed as a vector–Jacobian product (VJP) using reverse-mode automatic differentiation. We profile one guided-velocity evaluation over 100 runs with batch size one.

Across the tested set sizes, the VJP accounts for approximately 31%–39% of a guided-velocity evaluation on average. It therefore introduces a non-negligible but non-dominant overhead without requiring explicit Jacobian construction. End-to-end posterior-sampling cost additionally scales with the number of solver steps, as evaluated in Table 13.

Table 6: Peak GPU memory and runtime comparison between standard attention and FlashAttention for SFP with FaMA blocks. All measurements use batch size 1 and the same model architecture.

Task	Tokens	Standard Mem. (MiB)	Flash Mem. (MiB)	Mem. Red.	Standard Time (ms)	Flash Time (ms)
1D GP	256	560	420	21.4%	2.20	1.09
Wind field	2048	882	448	49.2%	5.57	3.22
NS system	4096	2632	454	82.8%	17.79	12.48

Table 7: Per-step cost of posterior guidance. Wall-clock times are reported in milliseconds per sample as mean $\pm$ standard deviation over 100 evaluations with batch size one.

Dataset	N	Full guidance	VJP	VJP fraction
GP	128	1.702 ± 0.659	0.575 ± 0.620	$31.3 \pm 7.6\%$
GP	256	1.966 ± 0.157	0.607 ± 0.561	$30.9 \pm 28.9\%$
WF	512	2.662 ± 0.224	1.014 ± 0.255	$37.7 \pm 6.4\%$
WF	1,024	3.673 ± 0.173	1.419 ± 0.261	$38.5 \pm 5.8\%$
WF	2,048	8.507 ± 0.597	3.290 ± 0.489	$38.7 \pm 3.4\%$

E DATASET

GP Synthetic Data. We follow the experimental setup as in (Dutordoir et al., 2023). For each input dimension d, we generate training functions by sampling from GP priors with kernels drawn from two families: squared exponential (RBF) and Matérn-$5/2$. Specifically, for each function, we first sample a kernel type $k \in \{k_{\mathrm{RBF}}, k_{\text{Matérn-}5/2}\}$ given as

$$k_{\mathrm{RBF}}(\boldsymbol{x}, \boldsymbol{x}') = \sigma^2 \exp\left(-\frac{\|x - x'\|_2^2}{2d\ell^2}\right),$$

$$k_{\text{Matérn-}5/2}(x, x') = \sigma^2 \left(1 + \frac{\sqrt{5}r}{\ell} + \frac{5r^2}{3\ell^2}\right) \exp\left(-\frac{\sqrt{5}r}{\ell}\right), \quad r = \|x - x'\|_2/\sqrt{d}.$$

Input locations are drawn uniformly from $[-2, 2]^d$, while the magnitude σ^2 and the length scale ℓ are sampled from log-normal distributions as $\sigma^2, \ell \sim \mathrm{LogNormal}(0, 0.5^2)$. For each input dimension d, we train a single model and evaluate its generalization on two downstream tasks: Tabular UCI regression and Bayesian optimization. The data-generation settings are reported in Table 8.

Table 8: GP Synthetic Data. Data generation settings for different input dimensions d. "Num points" denotes the total available points sampled per function instance.

	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$	$d=7$	$d=8$
total samples	51.2k	51.2k	1.24M	1.24M	1.24M	25.6M	25.6M
max num points (N)	256	256	256	512	512	2048	2048

Navier-Stokes Equation. We consider the two-dimensional incompressible Navier-Stokes equation in vorticity form, a standard model for studying instability and turbulence in fluid dynamics. After discarding the initial transient, the system evolves in a statistically stationary regime. We study the scalar vorticity field $\omega = \partial_x v_y - \partial_y v_x$, governed by

$$\frac{\partial \omega}{\partial t} + \boldsymbol{v} \cdot \nabla \omega = \nu \Delta \omega + g, \quad (x, y, t) \in [0, 2\pi]^2 \times (0, T],$$

where $\boldsymbol{v}$ is the incompressible velocity field, ν is the kinematic viscosity, and g denotes the vorticity forcing. We use the pseudo-spectral solver in the JAX-CFD toolbox to generate vorticity states (Kochkov et al., 2021). The initial conditions follow the procedure of Alieva et al. (2021). We set the maximum speed to $v_{\max} = 9.9$, with component-wise amplitudes $v_x, v_y = 7$ and viscosity $\nu = 0.02$.

Simulations are performed on a 64×64 grid with a time step of 2×10^{-3}, and snapshots are collected every 10 time steps. The dataset contains 100 training trajectories, 30 validation trajectories, and 20 test trajectories. Each trajectory contains 500 frames for 10 seconds, with a unique initial state.

Global Wind Field. We construct the wind-field dataset from the WeatherBench2 at `https://sites.research.google/gr/weatherbench/` Rasp et al. (2024) ERA5 archive on a 64×32 equiangular latitude–longitude grid with a 6-hourly resolution. From each timestamp, we extract the zonal and meridional wind vectors u, v at the 850 hPa level, yielding a 2-channel field with a shape $[64, 32, 2]$. Each snapshot is treated as one function instance $f\colon \mathbb{R}^2 \to \mathbb{R}^2$, mapping spherical coordinates (lat, lon) to (u, v). We use the period from 1989-01-01 to 2021-01-01 for training (46,752 snapshots) and from 2021-06-01 to 2022-01-01 for testing (856 snapshots). All metrics and training losses are area-weighted using normalized latitude weights $\tilde{w}_{ij} = w_i / \sum_{i,j} w_i$, where $w_i = \cos(\phi_i)$ and ϕ_i denote the latitude of the i-th grid row.

F EXPERIMENTS

F.1 UCI REGRESSION

We evaluate all methods on the UCI datasets in a zero-shot posterior-inference setting. Each model is pretrained exclusively on synthetic GP function samples with the corresponding input dimensionality, and no UCI observations are used for parameter optimization or fine-tuning. For each of 10 random seeds, we randomly partition each dataset into a 50% context set and a 50% query set. The context set is provided to the pretrained model as conditioning observations, while the query set is used only for evaluation. Feature- and target-normalization statistics are computed exclusively from the context set within each split and are subsequently applied to both the context and query sets. For the Energy Efficiency dataset, we use the heating-load response Y_1 as the scalar prediction target. For Auto MPG, we remove the six instances with missing horsepower values before constructing the splits, resulting in 392 usable instances. All methods are evaluated using exactly the same preprocessing and context–query partitions to account for variability due to data partitioning. This evaluation protocol follows common practice in probabilistic regression benchmarks and enables consistent comparison across datasets with different sample sizes and input dimensions. Table 9 summarizes the UCI benchmark datasets used in our evaluation.

Table 9: Summary of the UCI benchmark datasets used in the experiments. N denotes the total number of samples and d denotes the number of input features.

Task	Abbr.	N	d	ID	Domain
Airfoil Self-Noise	ASN	1,503	5		Aerodynamics
Concrete Compressive Strength	CCS	1,030	5		Civil Engineering
Auto MPG	AutoMPG	392	7		Automotive
Energy Efficiency	EE	768	8		Building Physics

F.2 BAYESIAN OPTIMIZATION

We evaluate posterior predictive quality in a sequential Bayesian optimization setting using standard benchmark functions from TRIESTE (Picheny et al., 2023). At the beginning of each run, an initial design is sampled uniformly from the function search space and evaluated by the exact objective. At each subsequent BO iteration, the current observations are used as the conditioning set for the surrogate model. The acquisition rule is discrete Thompson sampling (Gopalan et al., 2014): at every iteration, a finite candidate set is sampled from the search space, posterior samples are evaluated on these candidates, and the candidate(s) with the lowest sampled objective value are selected for evaluation. We use the known global minimum of each benchmark to compute simple regret,

$$r_t = \min_{i \le t} f(x_i) - f^\star,$$

and report regret trajectories aggregated over independent runs with different random seeds. The configuration for each task is specified in Table 10.

Table 10: Summary of the Bayesian Optimization tasks used in the experiments.

Task	**Dim**	**Num of Initials**	**Num of candidates**	**Num of batch size**	**Num of Steps**	**Total Num**
Branin2	2	20	128	1	180	200
Hartmann3	3	20	128	1	180	200
Ackley5	5	50	256	1	450	500
Trid7	7	50	256	1	450	500

F.3 METRICS

In this section, we include further details about the metrics used in Section 4.

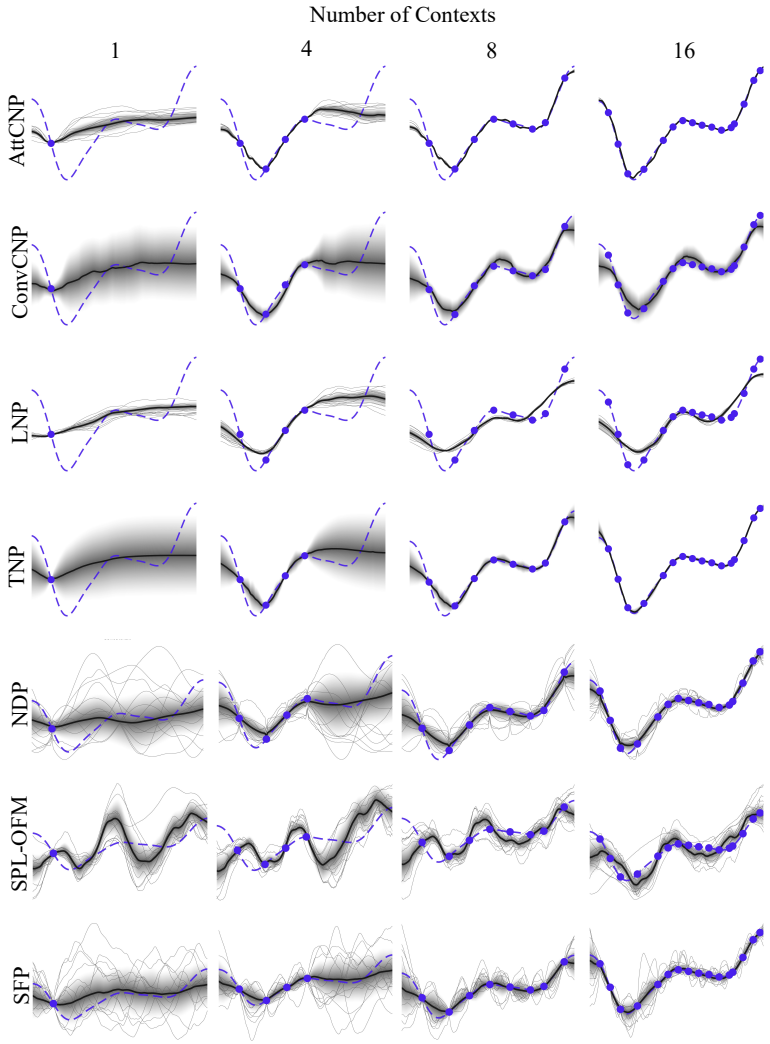


Figure 8: Gray curves denote posterior samples from different models. The empirical posterior mean is shown in black, together with sample quantiles. From left to right, the number of observed data points (blue dots) increases, and the posterior process becomes progressively closer to the underlying true function.

- **Normalized Root Mean Squared Error** (NRMSE) measures prediction accuracy normalized by the standard deviation of the ground truth, σ_y.

$$\text{NRMSE} = \frac{1}{\sigma_y}\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{\mu}_i)^2},$$

  where $\hat{\mu}_i$ is the predicted mean at index i. Lower is better.

- **Normalized Continuous Ranked Probability Score** (NCRPS): A proper scoring rule quantifying accuracy and calibration. Lower is better. We use $K = 20$ Monte Carlo samples $\{\hat{y}_i^{(k)}\}_{k=1}^{K}$ to characterize the predictive distribution. We estimate the NCRPS using the empirical form:

$$\text{NCRPS} = \frac{1}{N\sigma_y}\sum_{i=1}^{N}\left(\frac{1}{K}\sum_{k=1}^{K}|\hat{y}_i^{(k)} - y_i| - \frac{1}{2K^2}\sum_{k=1}^{K}\sum_{l=1}^{K}|\hat{y}_i^{(k)} - \hat{y}_i^{(l)}|\right).$$

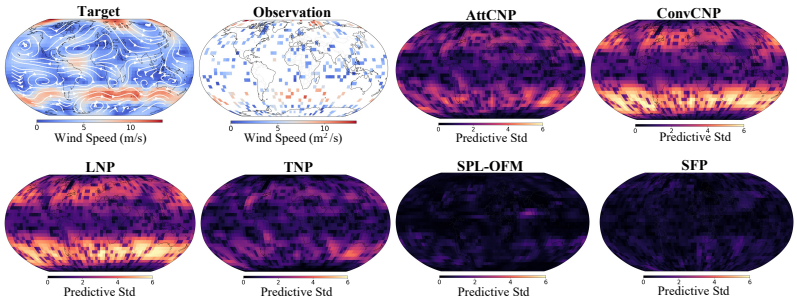


Figure 9: Predictive standard deviation in global wind field reconstruction.

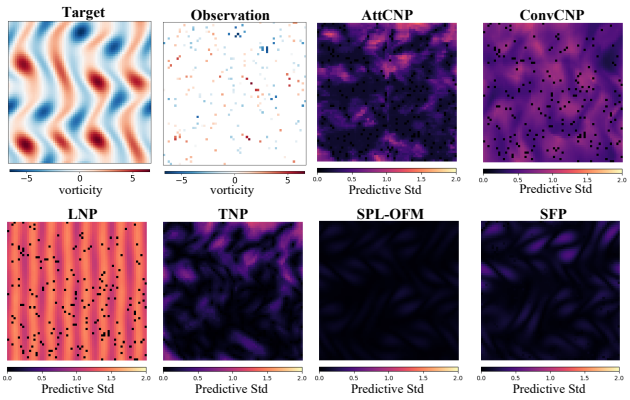


Figure 10: Predictive standard deviation in NS system reconstruction.

- **Maximum Mean Discrepancy** (MMD): A kernel-based distance between the predicted sample distribution and the reference distribution. Lower is better. Let $\{\hat{\boldsymbol{y}}^{(k)}\}_{k=1}^{K} \sim \hat{P}$ and $\{\boldsymbol{y}^{(m)}\}_{m=1}^{M} \sim P$ denote samples from the predicted and ground-truth distributions, respectively. We use an RBF kernel, with its length scale set by the median heuristic based on pairwise distances. The empirical squared MMD is

$$\widehat{\mathrm{MMD}}^2(\hat{P}, P) = \frac{1}{K(K-1)} \sum_{k \neq l} k\big(\hat{\boldsymbol{y}}^{(k)}, \hat{\boldsymbol{y}}^{(l)}\big) + \frac{1}{M(M-1)} \sum_{m \neq n} k\big(\boldsymbol{y}^{(m)}, \boldsymbol{y}^{(n)}\big) - \frac{2}{KM} \sum_{k=1}^{K} \sum_{m=1}^{M} k\big(\hat{\boldsymbol{y}}^{(k)}, \boldsymbol{y}^{(m)}\big).$$

- **Sliced Wasserstein Distance** (SWD): A distributional distance computed by projecting samples onto random one-dimensional directions and averaging the resulting Wasserstein distances. Lower is better. Let $\{\boldsymbol{\theta}_r\}_{r=1}^{R}$ be random unit vectors. The empirical SWD is

$$\mathrm{SWD}(\hat{P}, P) = \frac{1}{R} \sum_{r=1}^{R} W_1\Big(\{\langle \boldsymbol{\theta}_r, \hat{\boldsymbol{y}}^{(k)} \rangle\}_{k=1}^{K}, \{\langle \boldsymbol{\theta}_r, \boldsymbol{y}^{(m)} \rangle\}_{m=1}^{M}\Big),$$

  where W_1 denotes the one-dimensional Wasserstein-1 distance.

F.4 Empirical Evaluation of Marginal Consistency

SFP guarantees permutation invariance of its finite-dimensional distributions, but does not enforce exact marginal consistency (MC). Instead, approximate MC is encouraged during training by randomly sampling subsets. We empirically evaluate this property on the GP benchmark using nested input sets $X_S \subset X_L$, where $|X_S| = 128$ and $|X_L| = 256$. For each of 20 nested pairs, we draw

Table 11: Empirical marginal-consistency evaluation on the GP benchmark. Results are reported as mean ± standard deviation over 20 nested input-set pairs. Lower values indicate greater consistency.

Model	$10^2 \times$ MMD ↓	SWD ↓
Exact GP	2.87 ± 0.77	0.08 ± 0.02
NDP	11.92 ± 0.50	0.20 ± 0.02
SPL-OFM	$\mathbf{4.02 \pm 0.47}$	0.13 ± 0.01
SFP	4.15 ± 0.43	$\mathbf{0.11 \pm 0.01}$

Table 12: Sensitivity to the guidance scale on the WF validation set. Results are reported as mean ± standard deviation.

λ	NRMSE ↓	NCRPS ↓	CalErr ↓
0.05	0.37 ± 0.03	0.22 ± 0.02	0.05 ± 0.01
0.10	0.34 ± 0.02	0.21 ± 0.02	0.04 ± 0.01
0.30	0.29 ± 0.02	0.20 ± 0.02	0.03 ± 0.01
1.00	0.25 ± 0.01	0.14 ± 0.01	$\mathbf{0.02 \pm 0.01}$
3.00	$\mathbf{0.24 \pm 0.01}$	$\mathbf{0.13 \pm 0.01}$	$\mathbf{0.02 \pm 0.01}$
10.00	0.26 ± 0.02	0.16 ± 0.01	0.07 ± 0.01

2,048 samples both (i) directly on X_S, from $p_\theta(Y_S \mid X_S)$, and (ii) on X_L, subsequently restricting the samples to X_S, from $p_\theta(Y_L \mid X_L)\big|_{X_S}$. We compare the resulting empirical distributions using MMD and SWD, following Appendix F.3. Because both metrics are nonzero under finite-sample estimation, we include the exact GP, SPL-OFM, and NDP as references.

SFP and SPL-OFM are substantially closer to the finite-sample exact-GP reference than NDP. Among the learned models, SPL-OFM achieves the lowest MMD, whereas SFP achieves the lowest SWD and reduces MMD and SWD by approximately 65% and 45%, respectively, relative to NDP. These results provide empirical evidence that SFP approximately satisfies MC in this setting.

F.5 Sensitivity to Surrogate Variance and Guidance Scale

The surrogate variance r_t^2 is fixed by the flow parameterization. Under the linear optimal-transport path, $r_t^2 = \frac{(1-t)^2}{(1-t)^2+t^2}$, $t \in [0, 1)$, which decreases with flow time. In contrast, σ_y^2 denotes the observation-noise variance of the context values. We use the known observation variance in this paper. Following standard neural-process benchmarks, where context observations are treated as noiseless (Kim et al., 2018; Garnelo et al., 2018a; Dutordoir et al., 2023), we interpret σ_y^2 as a small nugget term for numerical stability (Williams & Rasmussen, 2006) and set $\sigma_y = 10^{-3}$.

We tune the guidance scale $\lambda \in \{0.05, 0.1, 0.3, 1.0, 3.0, 10.0\}$ separately for each task using validation NCRPS and keep the selected value fixed during testing. To assess posterior calibration, we additionally report the mean absolute coverage error

$$\text{CalErr} = \frac{1}{4} \sum_{c \in \{0.5, 0.8, 0.9, 0.95\}} \left| \widehat{C}_c - c \right|,$$

where $\widehat{C}_c$ is the empirical coverage of the c-level central posterior interval. Lower values indicate better calibration, with zero being optimal. Table 12 reports the validation results on the wind-field (WF) task.

Performance improves as the guidance strength increases from weak to moderate values, with the best results obtained for $\lambda \in [1, 3]$. Excessive guidance ($\lambda = 10$) degrades both predictive performance and calibration, suggesting over-constrained posterior inference. We therefore use $\lambda = 3$ for the WF test evaluation.

Table 13: Sensitivity to the number of sampling steps. Results are reported as mean ± standard deviation using batch sizes of 32 for GP and 8 for WF.

Dataset	Steps	Time (ms/sample)	NRMSE ↓	NCRPS ↓	CalErr ↓
GP	5	2.35 ± 0.02	0.44 ± 0.87	0.34 ± 0.55	0.19 ± 0.09
GP	10	4.58 ± 0.01	0.24 ± 0.11	0.19 ± 0.08	0.11 ± 0.05
GP	20	9.17 ± 0.01	0.13 ± 0.04	0.05 ± 0.05	0.08 ± 0.03
GP	50	22.75 ± 0.01	0.12 ± 0.05	0.05 ± 0.04	0.07 ± 0.03
GP	100	45.53 ± 0.01	0.05 ± 0.04	$\mathbf{0.03 \pm 0.02}$	0.05 ± 0.02
GP	200	91.08 ± 0.02	$\mathbf{0.05 \pm 0.02}$	0.04 ± 0.02	$\mathbf{0.05 \pm 0.01}$
WF	5	26.89 ± 0.21	0.53 ± 0.21	0.28 ± 0.13	0.22 ± 0.10
WF	10	53.40 ± 0.22	0.41 ± 0.09	0.23 ± 0.06	0.17 ± 0.05
WF	20	106.54 ± 0.19	0.33 ± 0.02	0.19 ± 0.02	0.14 ± 0.02
WF	50	265.57 ± 0.19	0.30 ± 0.01	$\mathbf{0.17 \pm 0.02}$	0.14 ± 0.02
WF	100	530.55 ± 0.20	0.28 ± 0.01	$\mathbf{0.17 \pm 0.01}$	$\mathbf{0.09 \pm 0.01}$
WF	200	1061.73 ± 0.20	$\mathbf{0.25 \pm 0.01}$	$\mathbf{0.17 \pm 0.01}$	0.10 ± 0.01

Table 14: Sensitivity to the spatial-scale parameter c on the WF validation set. Results are reported as mean ± standard deviation.

c	NRMSE ↓	NCRPS ↓
0.1	0.31 ± 0.03	0.23 ± 0.02
0.5	$\mathbf{0.23 \pm 0.01}$	$\mathbf{0.13 \pm 0.01}$
1.0 (used)	0.24 ± 0.01	$\mathbf{0.13 \pm 0.01}$
2.0	0.25 ± 0.02	0.15 ± 0.01
5.0	0.30 ± 0.01	0.19 ± 0.01

F.6 Ablation on the Number of Sampling Steps

The number of sampling steps controls the discretization accuracy of the sampling dynamics and directly affects both generation quality and computational cost. We evaluate sensitivity to the number of sampling steps on the GP and WF tasks. We vary the number of steps over $\{5, 10, 20, 50, 100, 200\}$, while keeping all other settings fixed, and evaluate 100 test cases using NRMSE, NCRPS, coverage calibration error, and sampling time.

Performance improves rapidly over the first 20-50 sampling steps. Further increases provide smaller and occasionally task-dependent gains, while sampling time grows approximately linearly. Calibration generally improves up to 100 steps, consistent with the behavior reported by Hamad & Rosenbaum (2025).

F.7 Sensitivity to the Spatial-Scale Parameter

The spatial-scale parameter c controls the range of spatial dependencies captured by the model. Our design reduces scale selection to this single parameter, and because all input coordinates are normalized to $[-2, 2]^d$, its value remains comparable across input dimensions. We select c from $\{0.1, 0.5, 1.0, 2.0, 5.0\}$ using validation NCRPS. Table 14 reports the results on the WF validation set.

Performance is relatively stable for $c \in [0.5, 2.0]$, with the best results obtained around $c \in [0.5, 1.0]$, whereas overly narrow or broad scale ranges degrade performance. This indicates that the method is moderately robust to c within a reasonable range. Our current design is limited to isotropic scales over normalized coordinates and does not capture dimension-specific anisotropy or arbitrary intrinsic geometries, which remain directions for future work.

Table 15: Single- versus multiscale spatial-bias ablation on the mixture-GP benchmark. Posterior metrics are reported over 100 test cases, while prior metrics are evaluated over five generation seeds. Lower is better.

Spatial bias	MMD ↓	SWD ↓	NRMSE ↓	NCRPS ↓
Single-scale	0.07 ± 0.02	0.12 ± 0.01	0.08 ± 0.04	0.05 ± 0.02
Multiscale	$\mathbf{0.04 \pm 0.01}$	$\mathbf{0.06 \pm 0.01}$	$\mathbf{0.05 \pm 0.04}$	$\mathbf{0.03 \pm 0.02}$

Table 16: Performance across five independent training seeds. For each trained checkpoint, metrics are first aggregated over the same set of test cases. We then report the mean and sample standard deviation of these training-level metrics across the five independently trained checkpoints.

Dataset	Method	MMD ↓	SWD ↓	NRMSE ↓	NCRPS ↓
Mix GPs	SPL-OFM	$\mathbf{0.03 \pm 0.01}$	0.08 ± 0.02	0.06 ± 0.02	0.03 ± 0.01
	NDP	0.05 ± 0.03	0.11 ± 0.05	0.10 ± 0.04	0.04 ± 0.02
	SFP	$\mathbf{0.03 \pm 0.01}$	$\mathbf{0.06 \pm 0.01}$	$\mathbf{0.04 \pm 0.01}$	$\mathbf{0.02 \pm 0.00}$
NS	SPL-OFM	0.15 ± 0.00	1.66 ± 0.01	$\mathbf{0.08 \pm 0.01}$	0.05 ± 0.00
	SFP	$\mathbf{0.14 \pm 0.01}$	$\mathbf{1.30 \pm 0.01}$	0.10 ± 0.01	$\mathbf{0.03 \pm 0.00}$
Wind field	SPL-OFM	0.23 ± 0.01	2.11 ± 0.04	0.31 ± 0.00	0.22 ± 0.00
	SFP	$\mathbf{0.22 \pm 0.01}$	$\mathbf{1.94 \pm 0.03}$	$\mathbf{0.27 \pm 0.01}$	$\mathbf{0.17 \pm 0.00}$

F.8 Single- versus Multiscale Spatial Bias

To isolate whether the improvement arises merely from introducing a spatial bias or specifically from its multiscale design, we compare two otherwise identical SFP models. The single-scale variant assigns the same spatial scale $\ell_h = 1$ to every attention head, whereas the multiscale variant uses the proposed log-spaced, head-specific scales with $c = 1$. All remaining architectural, training, and evaluation settings are held fixed. The multiscale variant improves all four mean metrics, reducing MMD, SWD, NRMSE, and NCRPS by approximately 43%, 50%, 38%, and 40%, respectively. Because the mixture-GP benchmark contains RBF and Mat'ern-$5/2$ kernels with heterogeneous length scales, these results directly support the benefit of assigning different spatial scales to different attention heads rather than using a single fixed scale.

F.9 Robustness Across Training Seeds

We assess sensitivity to training initialization using five independently trained checkpoints for each applicable generative model. We vary only the training seed while keeping the data-generation, model, and evaluation protocols fixed. For each checkpoint, we compute all metrics on the same evaluation cases. We then report the mean and sample standard deviation of the checkpoint-level metrics across the five training runs.

The comparative pattern remains stable across independently trained checkpoints. SFP achieves the best or tied MMD and SWD on all three benchmarks and the lowest NCRPS, whereas SPL-OFM obtains the lowest NRMSE on the NS task. These results indicate that the main comparative conclusions are robust to training initialization across the five evaluated seeds.

F.10 Comparison with Flow Matching Neural Processes

We additionally compare SFP with Flow Matching Neural Processes (FMNP) (Hamad & Rosenbaum, 2025) using its official implementation and GP-task configuration. We align the data and posterior-evaluation protocols with our main GP experiment. Because FMNP is trained as a context-conditioned predictive model rather than as a separate unconditional function prior, unconditional MMD and SWD are not applicable under this configuration. SFP and FMNP achieve comparable posterior performance: their NCRPS values have the same mean, while their NRMSE intervals overlap. The

Table 17: Comparison with FMNP on the mixture-GP benchmark. A dash indicates that the method does not define the corresponding unconditional prior under the evaluated configuration.

Method	MMD ↓	SWD ↓	NRMSE ↓	NCRPS ↓
FMNP	–	–	0.06 ± 0.03	0.03 ± 0.01
SFP	0.04 ± 0.01	0.06 ± 0.01	0.05 ± 0.04	0.03 ± 0.02

Table 18: Sampling and guidance hyperparameters used for the reported posterior results.

Task	Solver	Steps K	Guidance scale λ
Mix GPs	Euler	100	1.0
Navier–Stokes	Euler	100	3.0
Wind field	Euler	100	3.0
UCI regression	Euler	100	1.0
Bayesian optimization	Euler	100	1.0

distinction is therefore not a claim of uniformly superior conditional accuracy; rather, SFP additionally learns a reusable unconditional prior and supports explicit likelihood-based posterior inference.

F.11 Sampling Solver and Guidance Hyperparameters

Unless otherwise specified, we solve the SFP sampling ODE using the first-order explicit Euler method with $K = 100$ uniformly spaced steps. For posterior inference, we select the guidance scale from

$$\lambda \in \{0.05, 0.1, 0.3, 1.0, 3.0, 10.0\}$$

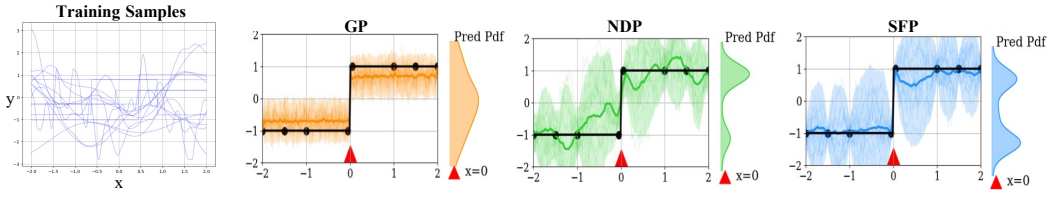
using validation NCRPS and keep the selected value fixed during testing. Context observations are treated as noiseless after output normalization, and we set $\sigma_y = 10^{-3}$ as a numerical-stability nugget. The selected guidance scales are summarized below.

F.12 Multimodal Posterior

Following Dutordoir et al. (2023), we evaluate SFP on multimodal posterior modeling. Unlike their setting, we construct the training data as a mixture of synthetic GP functions and step functions with diverse jump magnitudes and varying change points. This design induces discontinuous sample paths and stronger posterior ambiguity, providing a more challenging benchmark for evaluating multimodal uncertainty modeling. The step-function data are generated synthetically by sampling a change point $c \sim U[-1, 1]$ and a jump magnitude $h \sim U[-1, 1]$, and defining

$$f(x) = \begin{cases} -h, & x \le c, \\ h, & x > c. \end{cases}$$

We generate 100k such samples and combine them with the synthetic 1D GP samples described in Table 8. Example training functions are shown in Figure 11 (a). Figure 11 (b) compares the



(a) Training Samples (b) Predictive Posteriors

Figure 11: Multimodal posterior modeling. Figure (a) shows the training samples. Figure (b) shows fitted posterior distributions from a GP with a Matérn-2.5 kernel, NDP, and SFP, together with the predictive density at the change point $x = 0$.

posterior samples and the predictive density at the change point $x = 0$. Because the two conditioning observations lie close to the jump location, the posterior is highly ambiguous. The GP posterior with a Matérn-2.5 kernel tends to smooth across the discontinuity, producing noisy interpolation artifacts and an essentially unimodal predictive density, thereby failing to capture the two plausible function values induced by the step structure. NDP captures larger uncertainty around the transition region, but its samples are relatively inconsistent with the step functions, and the predictive density only weakly separates the modes. In contrast, SFP produces sharper posterior samples that align with the two discontinuous branches and yields a clearly bimodal predictive density at $x = 0$, indicating better recovery of the multimodal posterior structure.