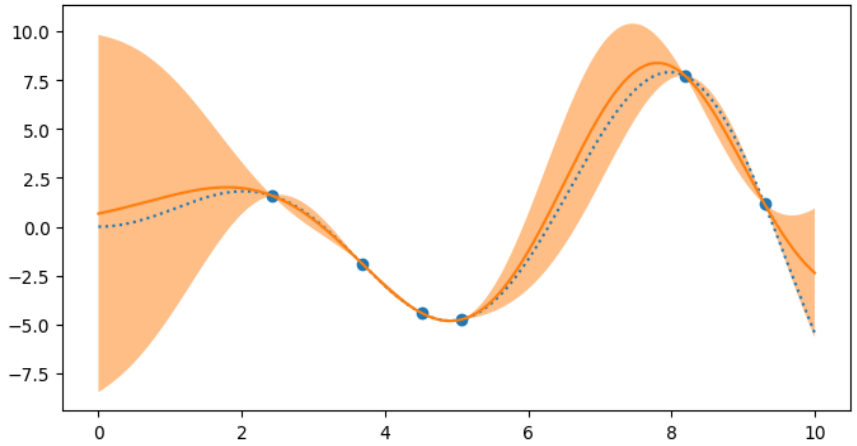
Stochastic Process Learning via Operator Flow Matching

Yaozhong Shi , Zachary E. Ross, Domniki Asimaki - Caltech

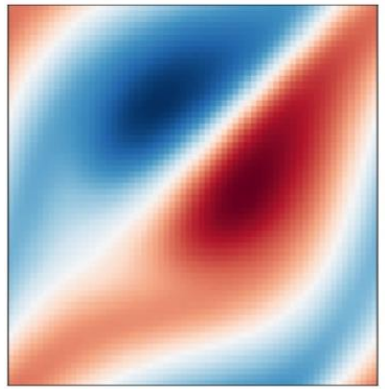
Kamyar Azizzadenesheli - Nvidia

Stochastic Process Learning

- Stochastic processes are foundational to many domains
- Serve as prior over functions, and provide density of ***any finite collections*** of points
- Many processes are not well described by Gaussian Processes (GP) -> we need more general stochastic process learning (SPL)



GP √



GP ×

General stochastic process regression

- A long-standing challenge kept researchers busy for decades
- Core difficulties :

(1) Learning a general stochastic-process prior from historical data with an expressive enough model

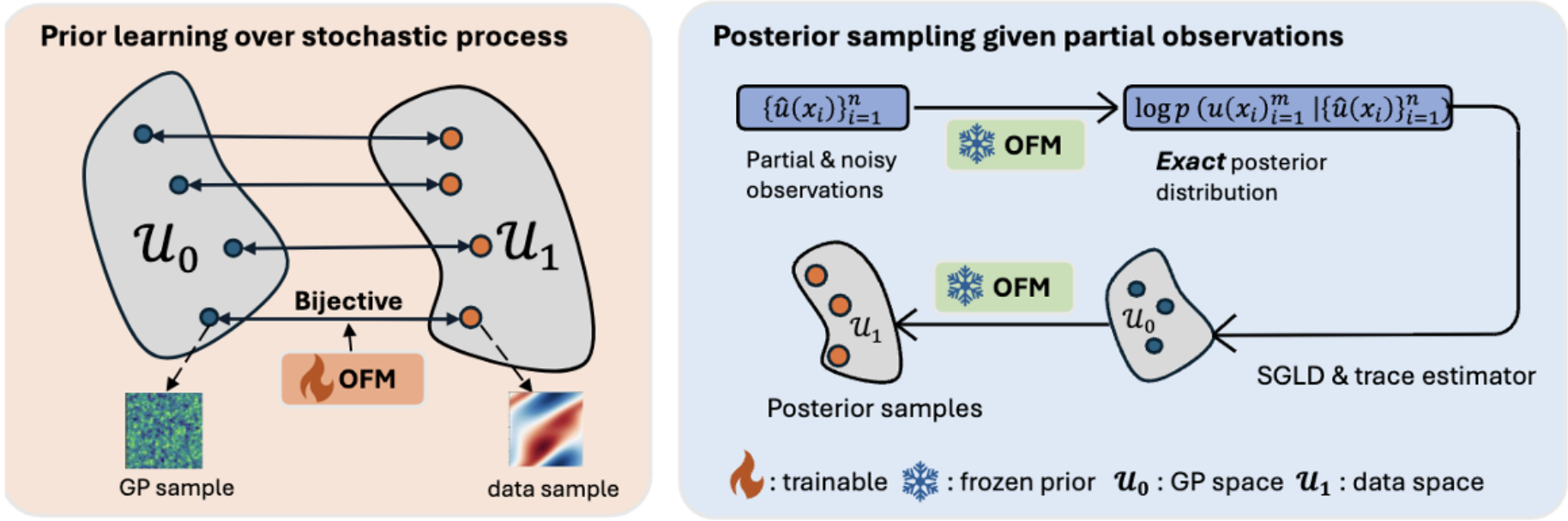
(2) Deriving both the exact posterior distribution and an efficient sampling scheme

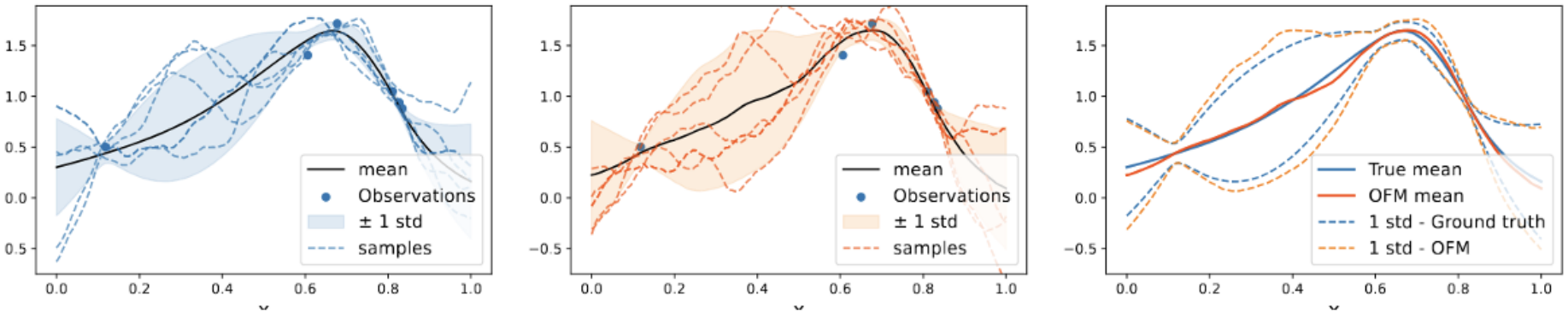
- Prior work :

(1) GPs : not expressive, limited to GP case

(2) Deep GPs & Neural Process (NPs) : expressive but not exact

Two-phase strategy





Prior Learning – Flow Matching in Hilbert Space

- Traditional flow matching :

(1) learn invertible mapping between $\mathrm{q}_0 = \mathcal{N}(0, I)$ and the target data distribution q_1

(2) Lebegue measure involved, no restriction to the modal architecture for learning the vector field

- Infinite-dimensional Flow Matching:

(1) learn invertible mapping between a Gaussian measure $\nu_0 = \mathcal{N}(0, C)$ and the target data measure ν_1

(2) Probability measure defined in Hilbert space involved, neural operator is is required for learning the vector field

(3) The covariance operator C is trace-class, (*white noise kernel is not trace-class*)

Generalizing FM to Stochastic Process

- Extend neural operator to maps between collections of points
- Generalizing FM to stochastic process is naturally induced from FM in Hilbert space
- Kolmogorov extension theorem (KET) is satisfied
- Intuitive explanation :

For any n and points $\{x_1, x_2, \dots, x_n\} \subset D$, the flow transports reference finite-dimensional distribution $p_0 = \mathcal{N}(0, K(\{x_1, .., x_n\})$ to the p_1, where p_1 is the marginalization of ν_1 on points $\{x_1, x_2, \dots, x_n\}$

Likelihood Estimation and Bayesian Universal Functional Regression

- Likelihood estimation

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

$O(m^2)$ *time complexity*

- Hutchinson Trace estimator (unbiased)

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

$O(m)$ *time complexity*

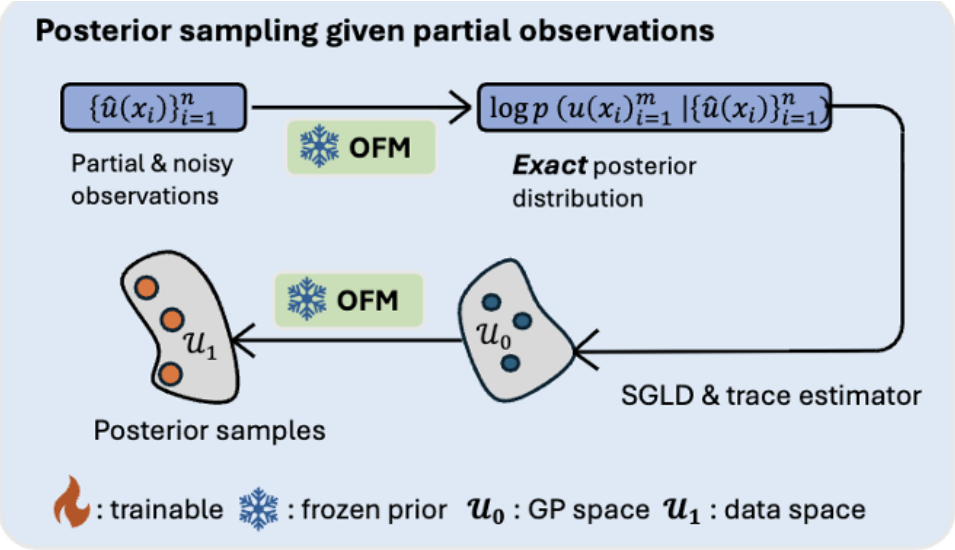
- Posterior distribution

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

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

Observations Likelihood Prior

Posterior Sampling with Stochastic Gradient Langevin Dynamic



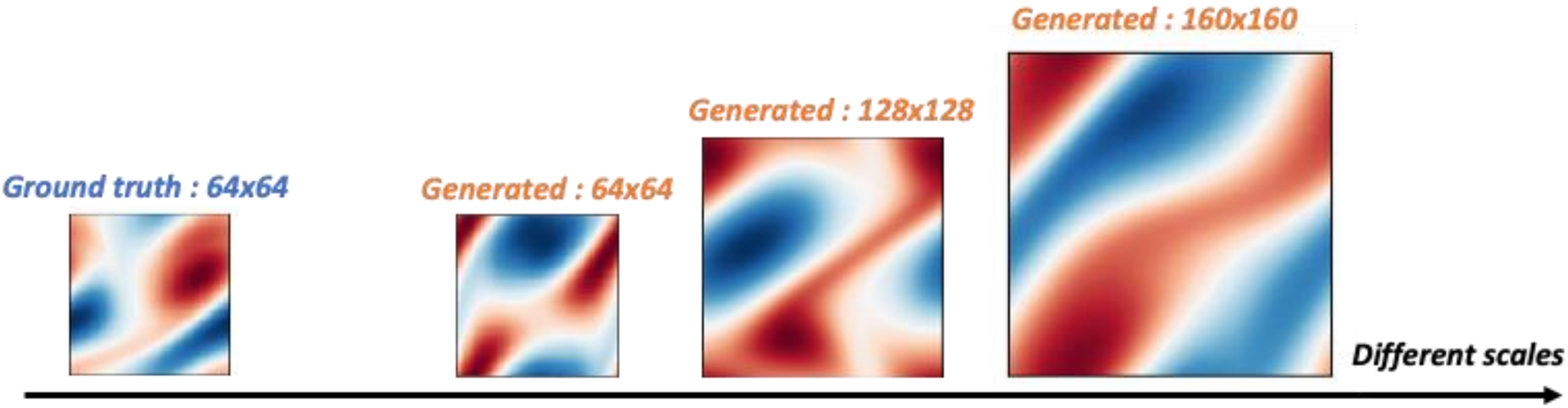
Algorithm 1 Posterior sampling with SGLD

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

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

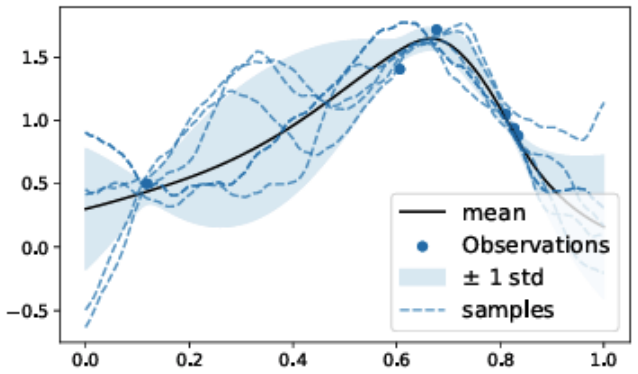
- Sample in the GP space $\mathcal{U}_0$, rather than the data sample $\mathcal{U}_1$
- Rely on the Hutchinson trace estimator, which requires many noise samples for divergence evaluation
- Requires integration over the entire ODE trajectory and can be GPU-memory intensive

Results : Zero-shot Generation with Learnt Prior

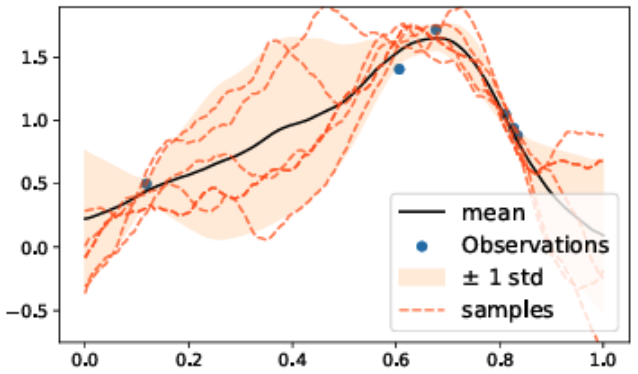


Results – Functional Regression Performance

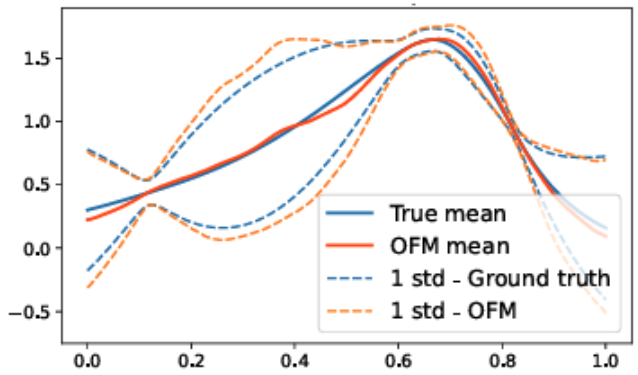
- Reproduce exact GP regression if prior is Gaussian, provide correct posterior for non-GP case



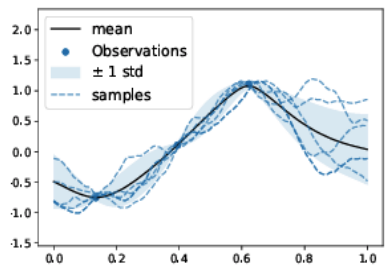
(a) Ground truth



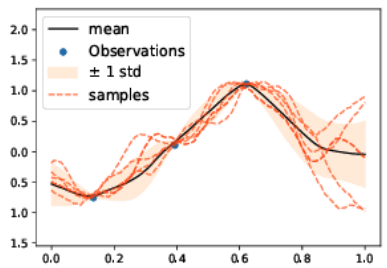
(b) OFM



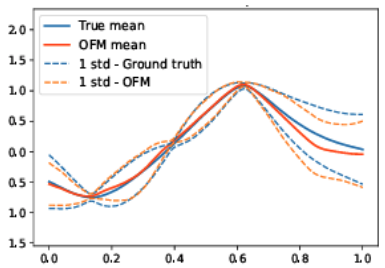
(c) Uncertainty comparison



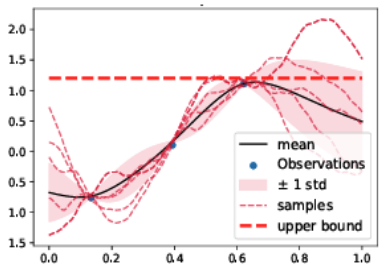
(a) Ground truth



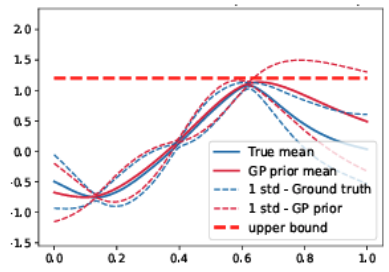
(b) OFM



(c) Uncertainty of OFM



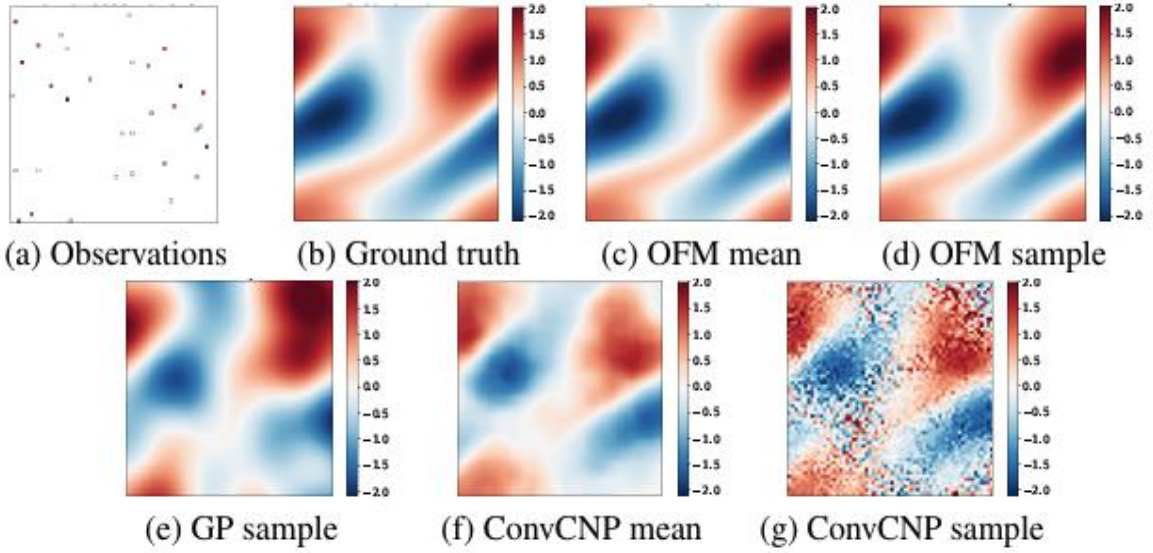
(d) GP prior

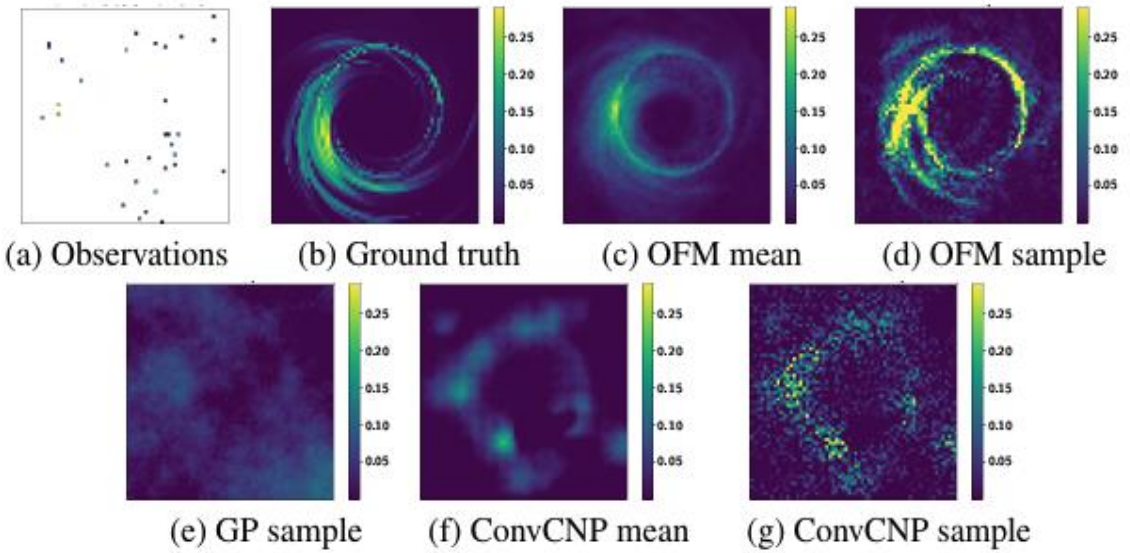


(e) Uncertainty of GP

Results – Functional Regression Performance

- Significantly better than baselines for complicated non-Gaussian regression cases





Results – Functional Regression Performance

- Performance against baslines

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

Summary and conclusion

- **Expressive prior over functions:** A neural operator learns a continuous flow that transports a reference GP to data-like functions, giving an explicit, tractable-density prior.
- **Function-level regression:** treat functions as first-class objects rather than mere pointwise values (unlike NPs), predictions are consistent across resolutions and arbitrary query sets.
- **Invertible & Bayesian:** The flow is invertible, enabling change-of-variables likelihoods and principled Bayesian regression with calibrated uncertainty from few observations.