

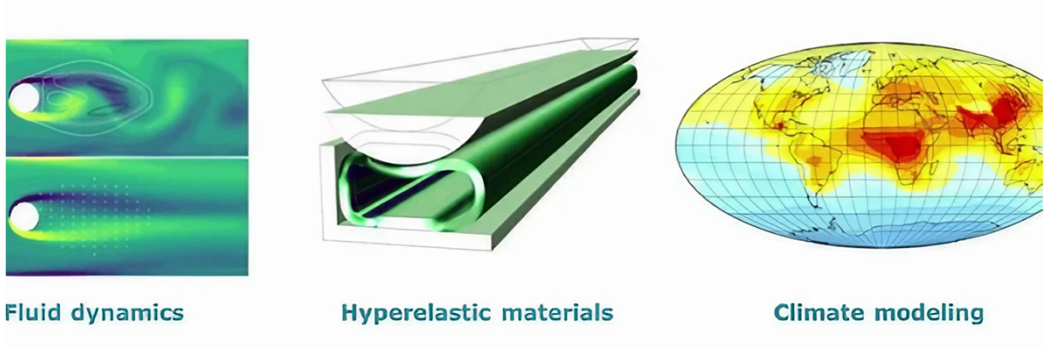

Uncertainty Quantification for Forward and Inverse Problems of PDEs via Latent Global Evolution

Tailin Wu[1*], Willie Neiswanger[2*], Hongtao Zheng[1*], Stefano Ermon[3], Jure Leskovec[3]

[1]School of Engineering, Westlake University
[2]Department of Computer Science, University of Southern California
[3] Computer Science Department, Stanford University

Motivation

Deep learning is rapidly becoming a go-to method in scientific and industrial applications, offering a faster alternative to traditional partial differential equation (PDE) solvers with speedups ranging from 10 to 1000 times.
However, one key challenge that **prevents their wide application** in science and industry is that practitioners do not know the uncertainty of the predictions, especially in places with critical decision making.



Difficulty

UQ for deep learning-based surrogate models for temporal PDEs presents unique challenges:

- **Generalization:** For deep learning models, it is hard to estimate the generalization error for *unknown* test inputs.
- **Uncertainty propagation:** For temporal PDEs, the surrogate model needs to perform autoregressive rollout. Thus, the uncertainty will accumulate.
- **Efficiency and scalability**: we need a fast and scalable method that can scale to millions of state dimension per time step, with fast rollout speed.

Related work

[1] Winovich, et al. "ConvPDE-UQ: Convolutional neural networks with quantified uncertainty for heterogeneous elliptic partial differential equations on varied domains." Journal of Computational Physics 394 (2019): 263-279.
[2] Zhu, Yinhao, et al. "Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data." Journal of Computational Physics 394 (2019): 56-81.

Past works have not addressed the problem.

Most works only deal with stationary PDEs that does not involve time:

(1) ConvPDE-UQ [1] learns a mapping from the elliptic PDE's parameter f to solution u, with UQ, where the PDE is given by:

$$\begin{cases} \Delta u = f & \text{in} \quad \Omega \\ u = 0 & \text{on} \quad \partial\Omega \end{cases}$$

(1) Similarly, [2] learns a mapping from elliptic PDE's parameter f to solution u, with UQ, without labeled data.

[1] Xu, Kailai, and Eric Darve. "ADCME: Learning spatially-varying physical fields using deep neural networks." arXiv preprint arXiv:2011.11955 (2020).

Related work

The ADCME [1] package provides tools for certain forward and inverse problems. Its UQ has severe limitations:

- It provides language for constructing a physics-informed neural network (PINN) as a Bayesian neural network, and then performing MCMC for UQ with these models.
- Limitation: their MCMC procedure is restrictive and not scalable (uses a random-walk Metropolis Hastings algorithm; requires specialized probabilistic programming framework to train). PINN cannot do autoregressive rollout and generalize beyond the training time range.

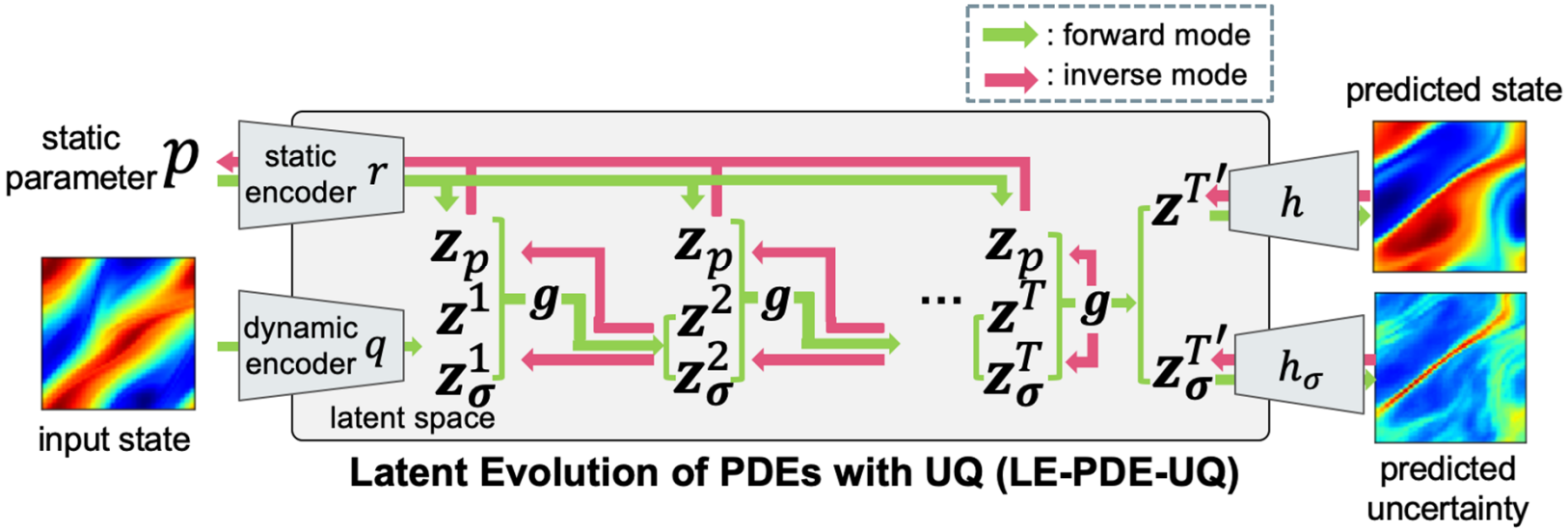
In Julia, there is also several libraries dealing with UQ, but none is doing neural surrogate models for temporal PDEs.

Our approach

We introduce Latent Evolution of PDEs with Uncertainty Quantification (LE-PDE-UQ), a method for learning deep learning-based surrogate models with **efficient and accurate uncertainty quantification**, for forward and inverse problems.

It evolves both the state of the system, and its uncertainty estimation of the state via respective latent vectors in latent space, and decode respectively to the state prediction and the uncertainty estimation.

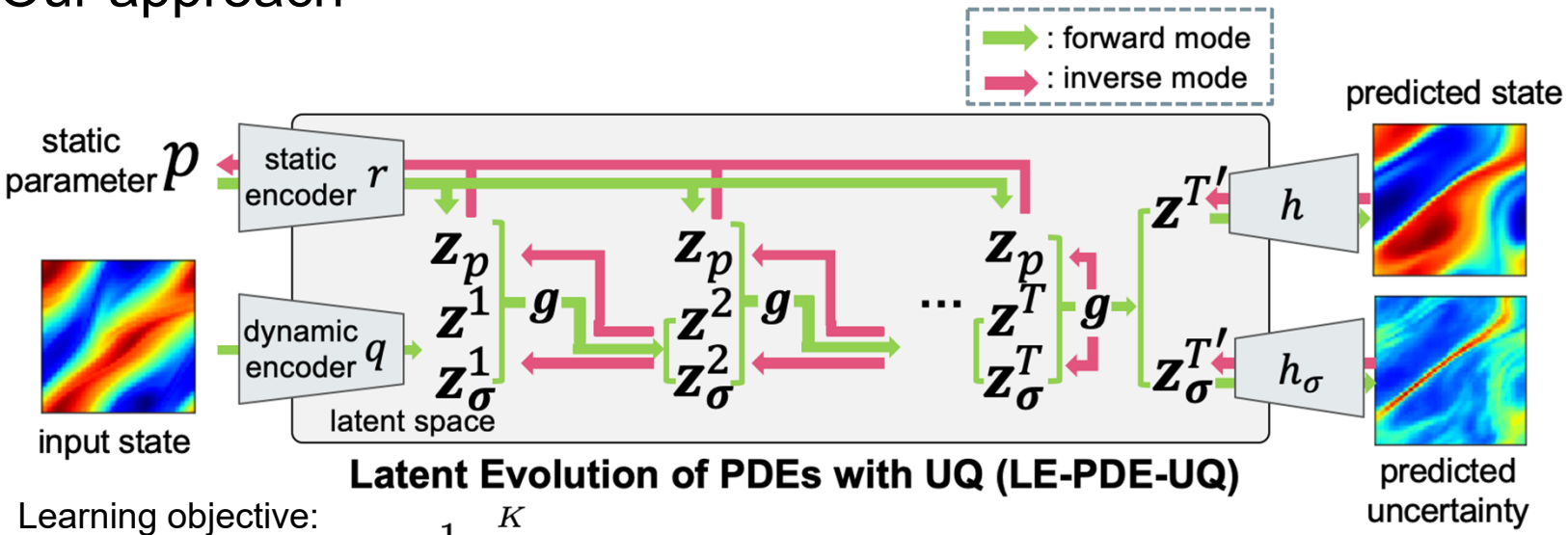
Our approach



- The latent evolution model g **deterministically** predicts both a global latent vector **z** that encodes the state, and a global latent vector $\mathbf{z}_\sigma$ that encodes the uncertainty. As needed, they decode respectively to the predicted state and predicted uncertainty.
- It allows evolution of state and **propagation of uncertainty** for long-term rollout.
- Also combine with Deep Ensembles [1] to capture epistemic uncertainty.

[1] Lakshminarayanan, Balaji, Alexander Pritzel, and Charles Blundell. *NeurIPS* (2017).

Our approach



Learning objective:

$$L = \frac{1}{K}\sum_{k=1}^{K}(L^k_{\text{multi-step}} + L^k_{\text{recons}} + L^k_{\text{consistency}}).$$

where

$$\begin{cases} L^k_{\text{multi-step}} = \sum_{m=1}^{M} \alpha_m \left[\frac{(\hat{U}^{k+m} - U^{k+m})^2}{2\hat{U}_\sigma^{k+m}} + \log \hat{U}_\sigma^{k+m}\right] & \text{(NLL in input space)} \\ L^k_{\text{recons}} = \ell(h(q(U^k)[z]), U^k) & \\ L^k_{\text{consistency}} = \sum_{m=1}^{M} \frac{||(g(\cdot, r(p))^{(m)} \circ q(U^k))[z] - q(U^{k+m})[z]||_2^2}{||q(U^{k+m})[z]||_2^2} & \text{(Consistency in latent space, for z)} \end{cases}$$

Experiments:

2D Navier-Stokes Equation with turbulence ($Re = 10^4$):

$$\partial_t w(t,x) + u(t,x) \cdot \nabla w(t,x) = \nu \Delta w(t,x) + f(x), \quad x \in (0,1)^2, t \in (0,T] \tag{10}$$

$$\nabla \cdot u(t,x) = 0, \quad x \in (0,1)^2, t \in [0,T] \tag{11}$$

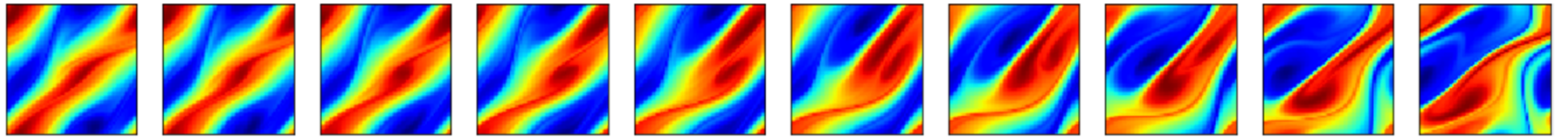
$$w(0,x) = w_0(x), \quad x \in (0,1)^2 \tag{12}$$

Here $w(t, x) = \nabla \times u(t, x)$ is the vorticity, $v \in R^+$ is the viscosity coefficient

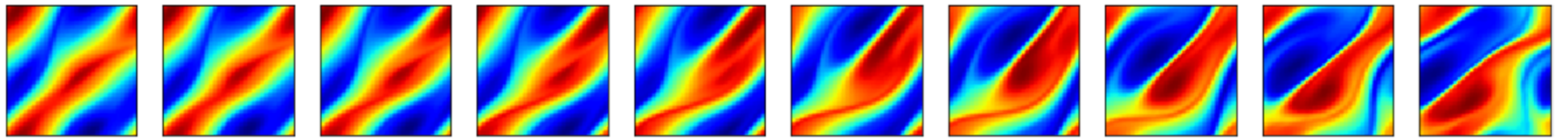
Experiment 1: forward problems

Input: initial state from t=1,2,...10

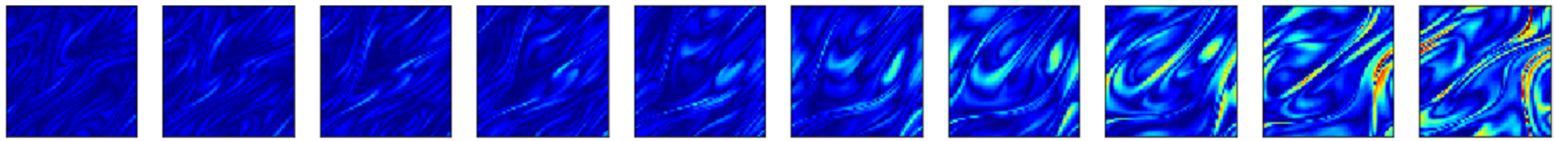
Ground-truth (t=11, 12, …20):



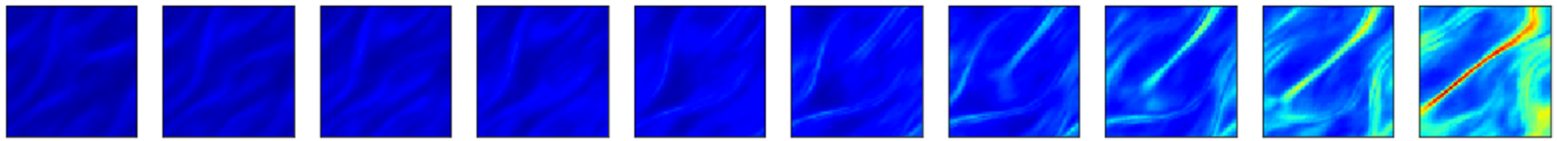
LE-PDE-UQ (ours) prediction:



Absolute error (scale: [0, 0.995]):

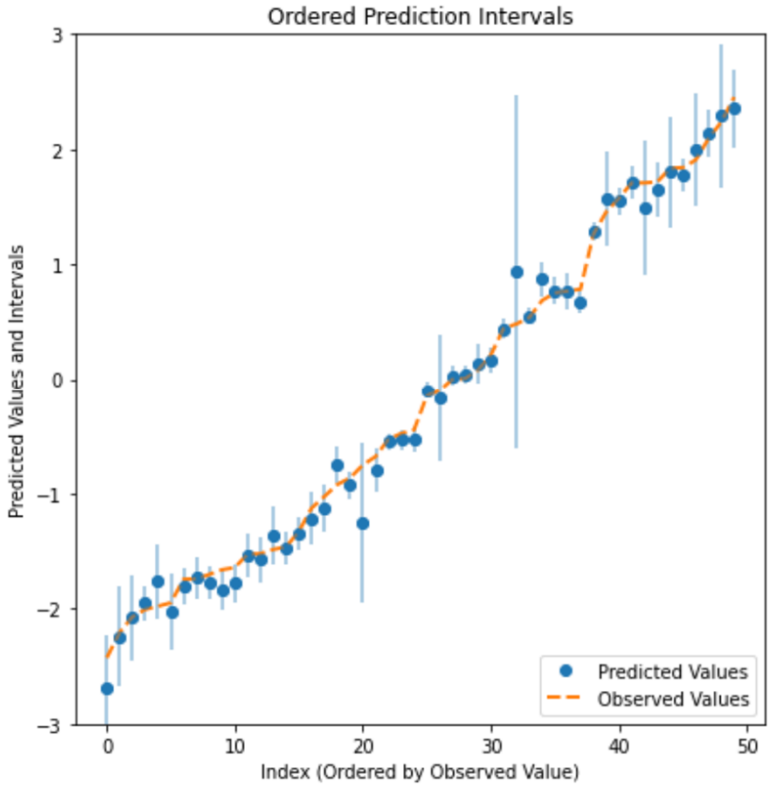


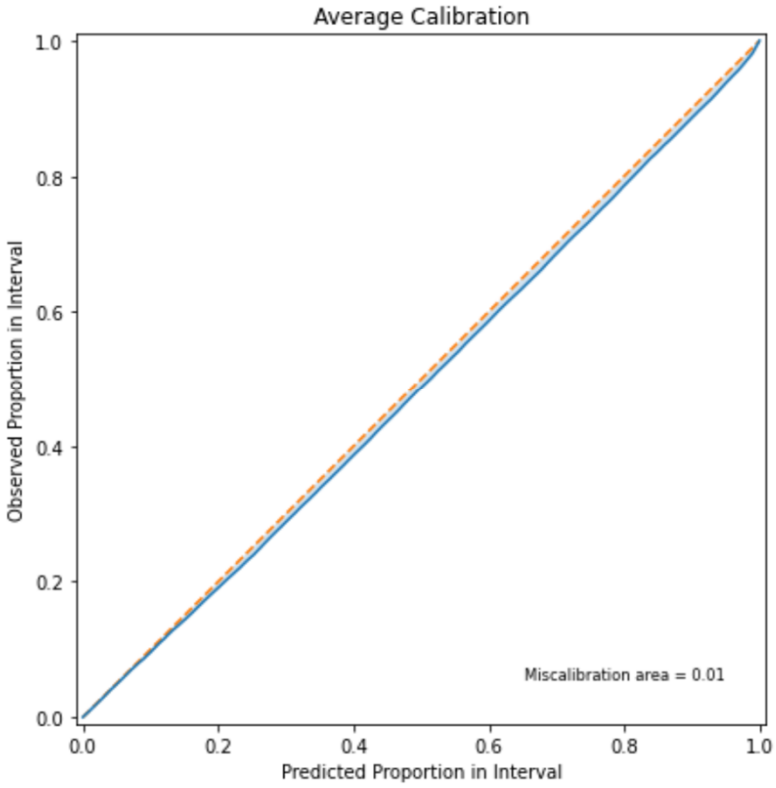
Predicted uncertainty (scale: [0, 0.995]):



Experiment 1: forward problems

Result of full model LE-PDE-UQ: Miscalibration area (MA): 0.01





Experiment 1: forward problems

Ensemble size: 10. **Bold**: best. Underscore: second best

MA: miscalibration area; MACE: mean absolute calibration error; RMSCE: Root mean squared calibration error

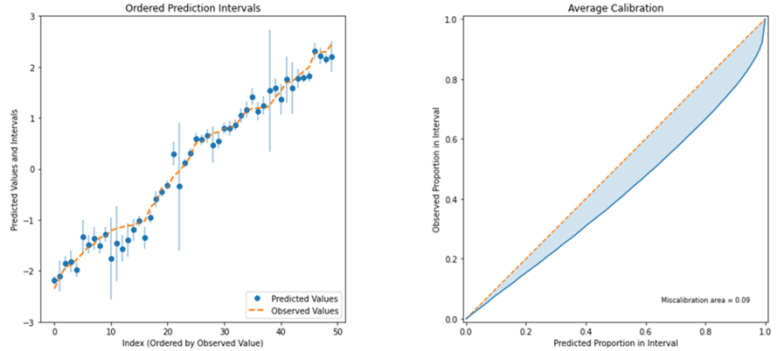
	MA	MACE	RMSCE	L2	MAE
Bayes layer with latent	0.0445	0.0440	0.0500	0.2345	0.2051
Bayes layer without latent	0.2381	0.2357	0.2665	0.2105	0.1830
Dropout, L2=0	0.1778	0.1760	0.1979	0.2079	0.1938
Dropout, L2=1e-5	0.1924	0.1905	0.2143	0.2092	0.1958
Dropout, L2=1e-4	0.2317	0.2294	0.2588	0.2458	0.2320
Dropout, L2=1e-3	0.3281	0.3248	0.3704	0.3534	0.3428
NoLatent (single, with σ)	0.1045	0.1035	0.1175	0.2053	0.1817
NoLatent (ensemble, without σ)	0.2118	0.2096	0.2355	0.1939	0.1657
NoLatent (ensemble, with σ)	0.0602	0.0596	0.0662	0.1939	0.1657
Latent (single, without σ)	-	-	-	**0.1890**	0.1613
Latent (single, with σ)	0.0576	0.0570	0.0649	0.2108	0.1811
Latent (ensemble, without σ)	0.1823	0.1805	0.2024	**0.1895**	**0.1608**
Latent (**ours**, ensemble, with σ)	**0.0142**	**0.0141**	**0.0160**	**0.1895**	**0.1608**

Main observations:

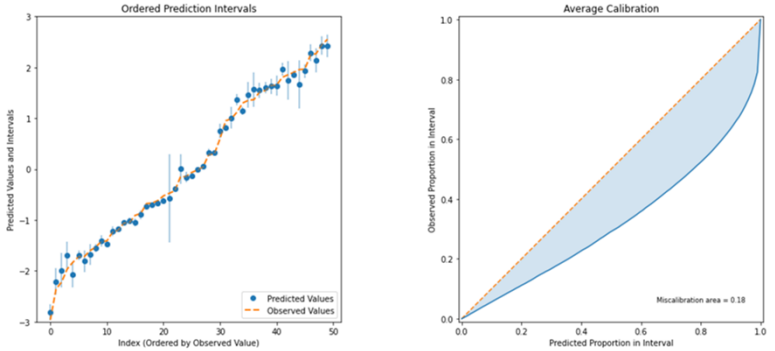
1. Our LE-PDE combined with ensemble attains the best performance in UQ and error.
2. Latent evolution significantly improves UQ: it can propagate uncertainty via long-term rollout
3. Deep Ensemble with deterministic prediction performs significantly worse in UQ

Experiment 1.1 Ablations

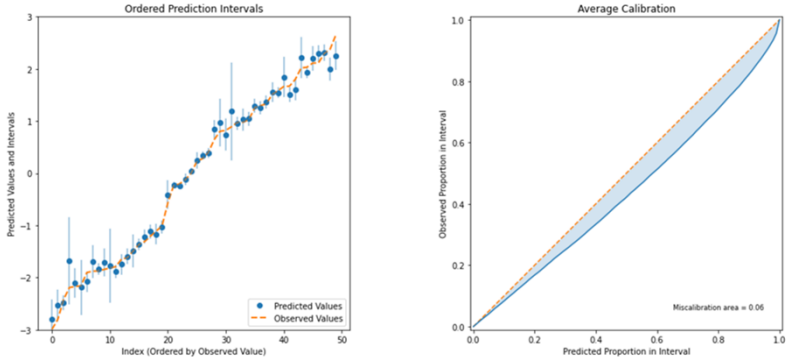
1. Forward problems **without latent**



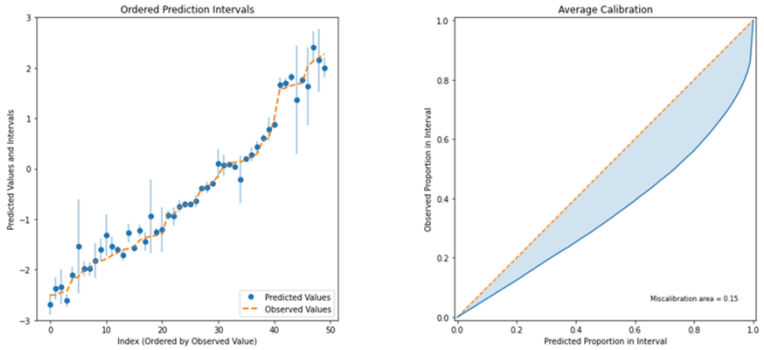
2. Forward problems **with deterministic**



3. **Latent full + L1**



4. Latent full + no z_σ^t



Experiment 1.2 latent vs. no latent

Autoregressive rollout:

	MA	MACE	RMSCE	L2	MAE
NoLatent (ensemble, with σ)	0.0602	0.0596	0.0662	0.1939	0.1657
Latent (**ours**, ensemble, with σ)	0.0142	0.0141	0.0160	0.1895	0.1608

To test how much the difference in performance is **uncertainty propagation** instead of different of models, we perform **teacher-forcing** (provide ground-truth as input) at each rollout step:

	MA	MACE	RMSCE	L2	MAE
NoLatent (ensemble, with σ)	0.0260	0.0258	0.0289	0.1670	0.1420
Latent (**ours**, ensemble, with σ)	0.0101	0.0100	0.0124	0.1562	0.1296

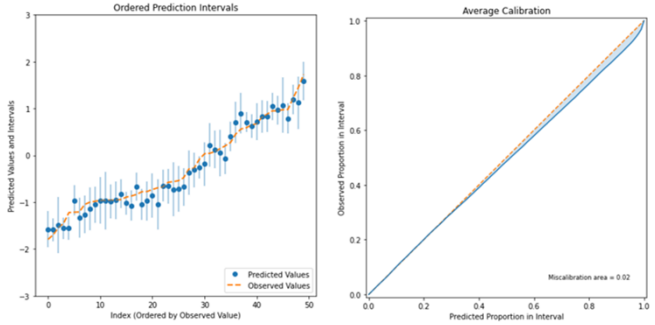
We see that difference in single step **cannot** account for the difference of MA for autoregressive rollout. The main benefit comes from the **uncertainty propagation** provided in our LE-PDE-UQ.

Experiment 2: inverse problem

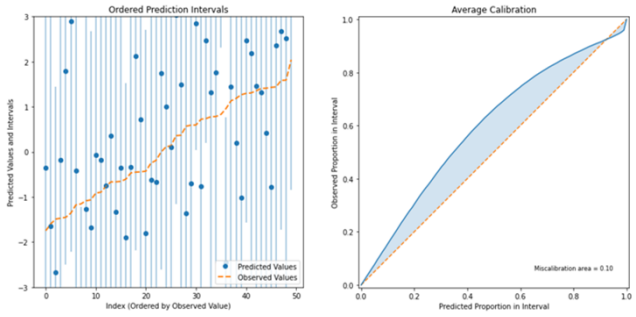
Predicting initial states (t=1,2,...9), given observation of future states (t=10,11,...20):

	MA	MACE	RMSCE	L2	MAE
NoLatent (ensemble, with σ)	0.0929	0.0920	0.1055	1.5255	1.2580
Latent (**ours**, ensemble, with σ)	0.0224	0.0222	0.0264	0.1863	0.1505

- No latent has significantly larger error, and larger miscalibration error.



1. LE-PDE-UQ



2. NoLatent

Conclusion

In this work, we introduced LE-PDE-UQ to quantify uncertainty in time-dependent PDEs with deep learning surrogates. It leverages latent vectors for precise predictions and robust uncertainty estimates in both forward and inverse problems. Excelling in long-term auto-regressive rollouts, LE-PDE-UQ outperforms major baselines, ensuring precision and stability in uncertainty quantification.