PerFlow: Physics-Embedded Rectified Flow for Efficient Reconstruction and Uncertainty Quantification of Spatiotemporal Dynamics

Hao Zhou[1], **Rui Zhang**[1*], **Han Wan**[1] and **Hao Sun**[1*]
[1]Gaoling School of Artificial Intelligence, Renmin University of China
{haozhou, rayzhang, wanhan2001, haosun}@ruc.edu.cn

Abstract

Reconstructing PDE-governed fields from sparse and irregular measurements is challenging due to their ill-posed nature. Deterministic surrogates are trained on dense fields that struggle with limited measurements and uncertainty quantification. Generative models, by learning distributions over spatiotemporal fields, can better handle sparsity and uncertainty. However, existing generative approaches enforce data consistency and PDE constraints simultaneously via sampling-time gradient guidance, resulting in slow and unstable inference. To this end, we propose **PerFlow**, a **P**hysics-**e**mbedded **r**ectified **Flow** for efficient sparse reconstruction and uncertainty quantification of spatiotemporal dynamics. PerFlow decouples observation conditioning from physics enforcement, performing guidance-free conditioning by feeding observations into rectified-flow dynamics while embedding hard physics via a constraint-preserving projection (e.g., incompressibility or conservation). Theoretically, we establish invariance guarantees to ensure that trajectories remain on the physics-consistent manifold throughout sampling. Experiments on various PDE systems demonstrate competitive reconstruction accuracy with sound physics consistency, while enabling efficient conditional sampling (e.g., 50 steps) and up to 320x faster inference than 2000-step guided diffusion baselines.

1 Introduction

Partial differential equations (PDEs) govern various dynamical systems in science and engineering [Holton and Hakim, 2013; Tu *et al.*, 2023; Strogatz, 2024]. While classical numerical solvers offer high accuracy and convergence guarantees, they require fine-grained spatiotemporal discretization, resulting in prohibitively high computational costs [Lapidus and Pinder, 1999; Tadmor, 2012]. To address this issue, neural surrogate models learn the solution operator between function spaces, aiming for rapid inference after training [Li *et al.*, 2020; Lu *et al.*, 2021; Zhang *et al.*, 2025]. Most existing surrogate models are deterministic and trained on gridded inputs (e.g., complete initial conditions), yet real-world measurement data is often sparse and irregular, rendering field reconstruction an ill-posed and highly challenging problem [Zhang *et al.*, 2015]. Furthermore, deterministic methods only provide point estimates and lack uncertainty estimation capability.

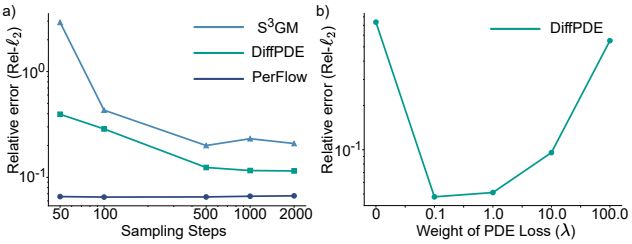

Figure 1: **(a)** Relative ℓ_2 error vs. sampling steps on Darcy (S^{3}GM, DiffPDE, PerFlow). **(b)** DiffPDE sensitivity on Poisson: Relative ℓ_2 error vs. PDE-loss weight λ.

Recent advances in generative models have enabled probabilistic surrogates for PDE systems. Diffusion [Ho *et al.*, 2020; Song *et al.*, 2020b] and flow [Lipman *et al.*, 2022] models learn a probability distribution over spatiotemporal fields, where stochastic sampling naturally supports uncertainty quantification. For forward modeling, generative PDE surrogates can capture multi-scale spectral features and quantify uncertainty, improving robustness to noise and distribution shift [Lippe *et al.*, 2023; Shysheya *et al.*, 2024; Li *et al.*, 2025]. More importantly, for sparse reconstruction, they can support conditionally guided generation, offering multiple high-fidelity plausible fields consistent with limited measurements [Huang *et al.*, 2024; Li *et al.*, 2024a].

Many generative methods achieve controllable generation by enforcing data consistency and physics constraints during sampling via gradient-based guidance [Li *et al.*, 2024a; Wang *et al.*, 2025]. However, this introduces two major issues. Firstly, it requires calculating observation-consistency gradients and physics gradients at each step of the sampling process, resulting in the slow inference speeds [Li *et al.*, 2024a]. Moreover, guidance tends to increase the required number of sampling steps and forces smaller step sizes

*Corresponding authors: Rui Zhang and Hao Sun.

for stable reconstructions. Secondly, it suffers from *sensitivity to hyperparameters*, one must carefully balance data-consistency and physics-informed terms to conform the prior distribution and guidance information, where reconstruction quality is highly sensitive to these weights in practice [Huang *et al.*, 2024]. Fig. 1 illustrates these issues on two representative guided diffusion baselines (DiffPDE [Huang *et al.*, 2024] and S^3GM [Li *et al.*, 2024a]).

To this end, we propose the **P**hysics-**e**mbedded **r**ectified **Flow** (**PerFlow**, shown in Fig. 2), an efficient conditional generative framework for sparse reconstruction and uncertainty quantification under partial observations. Unlike sampling-time guidance, PerFlow decouples observation conditioning from physics enforcement, avoiding gradient-based guidance at inference. For observed information, PerFlow incorporates sparse measurements as conditions instead of conducting iterative data-fidelity gradients during sampling. For physics information, PerFlow encodes conservation laws through a constraint-preserving projection, which can ensure the entire generation remains on the physics-consistent manifold. Finally, by adopting rectified flow, PerFlow supports fast few-step sampling, enabling efficient conditional sampling. In summary, we make the following contributions:

1. We propose **PerFlow**, a physics-embedded rectified-flow framework for sparse-field reconstruction and uncertainty quantification, enabling guidance-free conditional sampling.
2. We enforce hard physical constraints via constraint-preserving projection, and provide theoretical guarantees that sampling trajectories remain on the physics-consistent manifold throughout sampling.
3. We evaluate the performance of PerFlow across various PDE benchmarks. PerFlow achieves competitive reconstruction accuracy with remarkably lower physics error, while providing $\sim$320$\times$ speedup over guided diffusion baselines.

2 Related Work

2.1 Neural Surrogates for PDEs

With the development of deep learning, neural surrogate models have emerged as a data-driven approach to accelerating PDE simulation [Azizzadenesheli *et al.*, 2024; Zhang *et al.*, 2024; Bhaganagar and Chambers, 2025]. The main idea of neural surrogates is using a neural network to learn the mapping between function space, such as from PDE initial conditions to the corresponding solution. Compared with classical numerical methods, neural surrogate models do not need fine-grained spatiotemporal discretization and thus achieve remarkable speedup. DeepONet and its variants [Lu *et al.*, 2021; He *et al.*, 2024] use branch and trunk networks to encode the physical fields and spatiotemporal coordinates, respectively. Fourier neural operator and its variants [Li *et al.*, 2020; Rahman *et al.*, 2022; Xiong *et al.*, 2024; Li *et al.*, 2024b; Wan *et al.*, 2025] conduct function transformation in the frequency domain to achieve discrete invariance. Moreover, graph-based models [Eliasof *et al.*, 2021; Zeng *et al.*, 2024] and transformer-based models [Li *et al.*, 2023; Wu *et al.*, 2024] also extend neural surrogates to the irregular regime with their flexibility. However, most existing models use deterministic frameworks and conduct the training process across the complete fields, thus struggling with sparsity and uncertainty.

2.2 Generative Modeling of PDE Systems

Generative models offer a complementary approach to learning spatiotemporal systems by modeling *distributions* over trajectories rather than producing a single point estimate, such as variational autoencoders [Kingma and Welling, 2013], normalizing flows [Rezende and Mohamed, 2015], GANs [Goodfellow *et al.*, 2020], and diffusion models [Ho *et al.*, 2020; Song *et al.*, 2020a; Lu *et al.*, 2025] that sample via iterative denoising. Recently, continuous-time flow-based formulations, such as flow matching [Lipman *et al.*, 2022] and rectified flow [Liu *et al.*, 2022], learn a time-dependent velocity field and generate samples by integrating an ODE, reducing the number of function evaluations required for sampling. For physical systems, these models have been used for probabilistic forward simulation and for various downstream tasks [Xu *et al.*, 2022; Du *et al.*, 2024; Wang *et al.*, 2025; Hu *et al.*, 2025b], including parameter inference, system control, and sparse reconstruction [Wei *et al.*, 2024; Li *et al.*, 2024a; Huang *et al.*, 2024; Hu *et al.*, 2025a]. A common paradigm is to model the joint distribution of the full spatiotemporal fields and incorporate measurements and physics information through sampling-time guidance [Huang *et al.*, 2024]. However, this framework requires iterative denoising with per-step backpropagation, leading to *slow inference* and requiring more sampling steps or smaller step sizes for stable reconstructions. Moreover, these methods suffer from *hyperparameter sensitivity*, since reconstruction quality depends strongly on the relative weights used to balance observation and physics terms.

3 Method

3.1 Problem Setup

We study sparse reconstruction and uncertainty quantification for PDE-governed physical fields. Let $\Omega \subset \mathbb{R}^d$ be a spatial domain and $[0, T]$ a time interval. A broad class of systems can be written as

$$\mathcal{F}(\mathbf{x}; \boldsymbol{\theta}) = \mathbf{0}, \quad \mathbf{x}\big|_{\partial\Omega} = \mathbf{b}, \tag{1}$$

where $\mathbf{x}$ denotes the spatiotemporal field, $\boldsymbol{\theta}$ denotes PDE parameters, and $\mathbf{b}$ represents boundary conditions. In practice, only sparse and irregular measurements are available. We model sparse observations with a binary mask $\mathcal{M}$, where $\mathcal{M}_i = 1$ indicates an observed entry and $\mathcal{M}_i = 0$ otherwise:

$$\mathbf{y} = \mathcal{M} \odot \mathbf{x}. \tag{2}$$

Starting from sparse measurements $\mathbf{y}$, PerFlow infers the complete field $\mathbf{x}$ and represents uncertainty through samples from the conditional distribution

$$p(\mathbf{x} \mid \mathbf{y}, \mathcal{M}). \tag{3}$$

3.2 Overview of PerFlow

Most posterior-guided diffusion/flow approaches enforce data consistency and physics constraints *during sampling* via

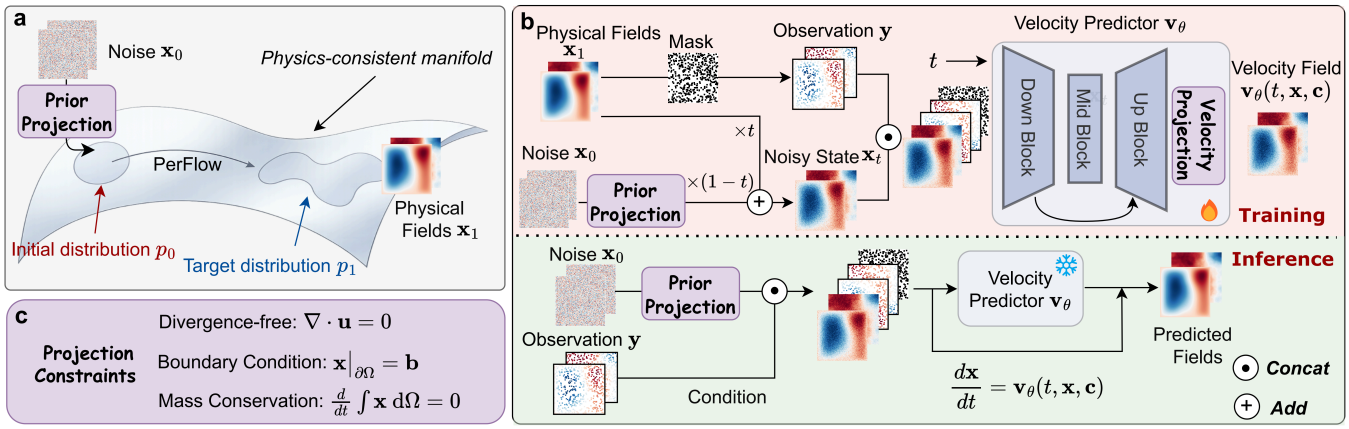

Figure 2: Overview of PerFlow. **(a)** PerFlow learns a rectified-flow transport between initial and target distributions on the physics-consistent manifold. **(b)** Training and inference pipelines. During training, sparse observations are formed by masking ground-truth fields, while prior-projected noise is combined with the target field using weights $(1-t)$ and t to construct $\mathbf{x}_t$. PerFlow learns a conditional velocity field $v_\theta(t, \mathbf{x}, \mathbf{c})$, followed by velocity projection. During inference, sampled noise is prior-projected and evolved by integrating $\frac{d\mathbf{x}}{dt} = v_\theta(t, \mathbf{x}, \mathbf{c})$. **(c)** Projection constraints embedded in PerFlow, including divergence-free constraints, boundary conditions, and mass conservation.

gradient-based guidance, resulting in slow inference and sensitivity to the guidance weights. To address these issues, PerFlow adopts rectified flow for efficient ODE-based sampling and decouples observation conditioning from physical guidance. PerFlow has the following key designs:

1. **Rectified flow for few-step inference.** Based on rectified flow, PerFlow samples by integrating a conditional ODE with a small number of function evaluations, enabling efficient reconstruction and uncertainty quantification.
2. **Guidance-free conditioning.** We regard sparse reconstruction as *conditional generation* and explicitly provide the sparse measurements as conditions, avoiding sampling-time data-fidelity gradients.
3. **Constraint-preserving projection.** We use *constraint-preserving projection* to ensure that trajectories remain on the physics-consistent manifold throughout sampling.

3.3 Rectified Flow Preliminaries

Rectified flow (RF) [Liu *et al.*, 2022] learns a continuous-time transport that maps a simple base distribution to a target conditional data distribution. Let p_0 be a base distribution and $p_1(\cdot \mid \mathbf{c})$ the target conditional distribution given $\mathbf{c}$. RF learns a time-dependent velocity field predictor $\mathbf{v}_\theta(t, \mathbf{x}, \mathbf{c})$ and generates samples by integrating

$$\frac{d\mathbf{x}}{dt} = \mathbf{v}_\theta(t, \mathbf{x}, \mathbf{c}), \qquad t \in [0, 1]. \tag{4}$$

Training is based on flow matching along an interpolation path. Given $\mathbf{x}_1 \sim p_1(\cdot \mid \mathbf{c})$ and $\mathbf{x}_0 \sim p_0$, we construct

$$\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\mathbf{x}_1, \qquad t \sim \mathcal{U}(0, 1), \tag{5}$$

and use the rectified-flow target velocity

$$\mathbf{v}^\star = \mathbf{x}_1 - \mathbf{x}_0. \tag{6}$$

Compared to diffusion-based sampling, RF admits efficient ODE solvers and typically requires fewer function evaluations, which is crucial for fast PDE reconstruction.

3.4 Conditioning on Sparse Observations

Unlike posterior-guided diffusion/flow methods that enforce observation consistency via sampling-time gradients, PerFlow casts sparse reconstruction as *conditional generation*. The observation information is incorporated through the condition input, enabling guidance-free inference without per-step backpropagation.

Given sparse measurements $\mathbf{y}$ and mask $\mathcal{M}$, we use a mask-aware condition

$$\mathbf{c} = \mathrm{concat}\big(\mathbf{y}, \mathcal{M}\big), \tag{7}$$

where $\mathcal{M}$ is broadcast to match channels if needed. We then learn a conditional velocity field $\mathbf{v}_\theta(t, \mathbf{x}, \mathbf{c})$ parameterized by a U-Net backbone. Training and inference procedures are detailed in Sec. 3.6.

3.5 Constraint-preserving Projection

A central design of PerFlow is to restrict the rectified-flow dynamics to a *physics-consistent manifold* that encodes hard PDE priors (Fig. 2**a**). Instead of injecting physics through gradient guidance, we propose the constraint-preserving projection to enforce essential constraints into the model architecture and sampling pipeline so that generated fields satisfy them by construction.

Where physics is embedded. PerFlow enforces hard constraints at two points: (i) the *physics-embedded prior*, which is obtained by applying a *prior projection* to the sampled noise, thereby yielding a physics-feasible initial state $\mathbf{x}(0)$; and (ii) the *constraint-preserving velocity projection*, where the learned velocity field is mapped to satisfy the corresponding constraints required by each ODE updates. We implement these designs using a set of physics-embedded operators, whose underlying physical constraints are summarized in Fig. 2**c** for clarity.

(1) Divergence-free. For 2D incompressible flow, we enforce incompressibility with a stream-function construction,

where we represent the velocity field $\mathbf{u} = (u, v)$ with a scalar stream function ψ:

$$u = \partial_y \psi, \qquad v = -\partial_x \psi, \tag{8}$$

which satisfies $\nabla \cdot \mathbf{u} = 0$ by construction. For the *physics-embedded prior*, we sample ψ from a Gaussian random field, and project it to a divergence-free initial state $\mathbf{u}(0) = \nabla^{\perp}\psi$. For the *constraint-preserving velocity projection*, the network predicts ψ, and we map it to (u, v) to obtain divergence-free velocity updates.

(2) Dirichlet boundary conditions. Let $\mathcal{B}$ be a binary boundary indicator (1 on $\partial\Omega$, 0 otherwise) and $\mathbf{b}$ represents boundary conditions. In detail, we replace the boundary values and keep interior entries unchanged to encode Dirichlet boundary conditions, which can be achieved via the operator Π_{bc} as follows:

$$\Pi_{\mathrm{bc}}(\tilde{\mathbf{x}}) = \mathcal{B} \odot \mathbf{b} + (1 - \mathcal{B}) \odot \tilde{\mathbf{x}}. \tag{9}$$

During ODE integration, we also keep boundary values fixed by applying the same boundary mask to the velocity, i.e., we set boundary updates to zero using $(1 - \mathcal{B}) \odot (\cdot)$.

(3) Global conservation. For scalar conserved quantities, we enforce a fixed total mass using the mean-correction operator Π_{mass}, which shifts the field to match the target mass:

$$\Pi_{\mathrm{mass}}(\tilde{\mathbf{x}}) = \tilde{\mathbf{x}} - \mathrm{mean}_{\Omega}(\tilde{\mathbf{x}}) + \frac{m_0}{|\Omega|}, \tag{10}$$

where m_0 represents the total mass. For velocity updates, we apply the corresponding zero-mean correction so that the ODE updates do not change the total mass. In practice, we apply Π_{mass} when sampling the physics-embedded prior, and apply the mean-correction to the predicted velocity field whenever this constraint is required.

By encoding these operators into both the physics-embedded prior and the constraint-preserving velocity projection, PerFlow starts from a physics-feasible state and performs rectified-flow sampling directly on the physics-consistent manifold without sampling-time physics guidance. Furthermore, we provide theoretical guarantees for the continuous-time dynamics in Sec. 4, where we prove that the physical constraint satisfaction can be maintained throughout sampling.

3.6 Training Objective and Conditional Sampling

We introduce PerFlow's training and inference procedures with sparse observations (Fig. 2**b**). The complete algorithmic procedure is provided in Appendix 2.

Training (Conditional RF with physics embedding). During training, we take a physical field $\mathbf{x}_1$ from the training set and sample a masking $\mathcal{M}$ to simulate sparse measurements $\mathbf{y} = \mathcal{M} \odot \mathbf{x}_1$, thus forming the condition $\mathbf{c} = \mathrm{concat}(\mathbf{y}, \mathcal{M})$. After that, we sample a random noise $\tilde{\mathbf{x}}_0$ from an initial distribution p_0 and project it to a physics-consistent initial state $\mathbf{x}_0 = \Pi(\tilde{\mathbf{x}}_0)$ using the projection operators in Sec. 3.5. We then construct $\mathbf{x}_t$ and $\mathbf{v}^{\star}$ as in Eqs. 5-6, and minimize

$$\mathcal{L}(\theta) = \mathbb{E}\Big[\big\|\mathbf{v}_\theta(t, \mathbf{x}_t, \mathbf{c}) - \mathbf{v}^{\star}\big\|_2^2\Big]. \tag{11}$$

In our implementation, $\mathbf{v}_\theta$ is made physics-consistent by constraint-preserving projection (Sec. 3.5), thus the learned dynamics is restricted to the physics-consistent manifold automatically.

Inference (Sparse reconstruction and uncertainty quantification). Given sparse observations $(\mathbf{y}, \mathcal{M})$, we set $\mathbf{c} = \mathrm{concat}(\mathbf{y}, \mathcal{M})$ to form a condition. We sample K i.i.d. physics-consistent initial states and integrate the conditional ODE in Eq. 4 from $t = 0$ to 1 using a few-step solver to obtain samples $\hat{\mathbf{x}}^{(k)} = \mathbf{x}^{(k)}(1)$. After that, we can estimate the uncertainty from the ensemble $\{\hat{\mathbf{x}}^{(k)}\}_{k=1}^{K}$ via posterior mean and variance.

4 Theory

This section states theoretical guarantees for PerFlow's hard-constraint embedding. After discretization, many physics priors (e.g., conservation and boundary conditions) can be written as affine linear constraints

$$\mathcal{S} \;=\; \{\mathbf{x} \in \mathbb{R}^d :\; A\mathbf{x} = \mathbf{p}\}, \tag{12}$$

where A stacks the corresponding constraint operators and $\mathbf{p}$ encodes prescribed values. PerFlow generates samples by integrating the conditional ODE

$$\frac{d\mathbf{x}}{dt} = \mathbf{v}_\theta(t, \mathbf{x}, \mathbf{c}), \tag{13}$$

where $\mathbf{c}$ is the observation condition. By design, PerFlow obtains a feasible initial state $\mathbf{x}(0) \in \mathcal{S}$ via prior projection and uses a constraint-preserving velocity field (Sec. 3.5). The following theorem formalizes the resulting trajectory-level constraint preservation.

Theorem 1 (Constraint invariance of continuous dynamics)**.** *Assume the feasible set is $\mathcal{S} = \{\mathbf{x} : A\mathbf{x} = \mathbf{p}\}$. Assume $\mathbf{v}_\theta(t, \mathbf{x}, \mathbf{c})$ is continuous in t and locally Lipschitz in $\mathbf{x}$ so that the ODE* (13) *admits a unique solution on $[0, 1]$. If (i) $\mathbf{x}(0) \in \mathcal{S}$ and (ii) the velocity field is tangent to $\mathcal{S}$, i.e.,*

$$A\,\mathbf{v}_\theta(t, \mathbf{x}, \mathbf{c}) = \mathbf{0} \quad \textit{for all } t \in [0, 1] \textit{ and all } \mathbf{x} \in \mathcal{S}, \tag{14}$$

then the solution satisfies $\mathbf{x}(t) \in \mathcal{S}$ for all $t \in [0, 1]$.

In PerFlow, we parameterize the velocity field in a constrained form (e.g., stream-function for incompressibility) or use an exact projection to the network output to enforce the corresponding constraints (mean correction for mass conservation). As a result, for any feasible state $\mathbf{x} \in \mathcal{S}$, the effective velocity satisfies $A\,\mathbf{v}_\theta(t, \mathbf{x}, \mathbf{c}) = \mathbf{0}$, which is exactly the tangency condition in Eq. 14. Therefore, with prior-projected initialization and constraint-preserving velocity projection, physical constraint satisfaction can be maintained throughout the entire sampling (detailed in Appendix 1). This trajectory-level guarantee differs from sampling-time guidance, which encourages physical consistency through additional correction terms during sampling but does not enforce feasibility for every ODE update. For nonlinear manifolds $\mathcal{M} = \{\mathbf{x} : g(\mathbf{x}) = 0\}$, trajectory invariance would require $J_g(\mathbf{x})\mathbf{v}_\theta(t, \mathbf{x}, \mathbf{c}) = 0$, where $J_g(\mathbf{x})$ is the Jacobian of g at $\mathbf{x}$; we leave this extension for future work.

PDE	Metric	UNet	FNO	FFNO	DiffPDE-50	S^{3}GM-50	PerFlow	DiffPDE-2000	S^{3}GM-2000
Burgers	Rel-ℓ_2	1.86×10^{-1}	1.23×10^{-1}	1.93×10^{-1}	1.45×10^{0}	6.99×10^{0}	$\mathbf{8.63 \times 10^{-2}}$	9.39×10^{-2}	1.87×10^{-1}
	Rel-ℓ_1	1.33×10^{-1}	8.09×10^{-2}	1.49×10^{-1}	1.47×10^{0}	6.81×10^{0}	$\mathbf{6.84 \times 10^{-2}}$	5.92×10^{-2}	1.60×10^{-1}
	Phys-Err	2.03×10^{-4}	4.62×10^{-5}	4.23×10^{-3}	1.19×10^{-4}	7.49×10^{-3}	$\mathbf{4.47 \times 10^{-11}}$	1.53×10^{-4}	1.04×10^{-4}
	Time	35.88	29.02	29.97	4.47	3.33	**0.79**	191.90	145.60
Poisson	Rel-ℓ_2	2.46×10^{-1}	1.66×10^{-1}	3.66×10^{-1}	9.16×10^{-1}	3.90×10^{1}	$\mathbf{5.02 \times 10^{-2}}$	5.12×10^{-2}	7.70×10^{-2}
	Rel-ℓ_1	1.97×10^{-1}	1.34×10^{-1}	3.23×10^{-1}	9.23×10^{-1}	3.96×10^{1}	$\mathbf{4.63 \times 10^{-2}}$	4.49×10^{-2}	7.11×10^{-2}
	Phys-Err	1.13×10^{-2}	4.13×10^{-2}	2.23×10^{-2}	8.37×10^{-3}	6.88×10^{1}	$\mathbf{0.00 \times 10^{0}}$	7.21×10^{-2}	6.21×10^{-2}
	Time	35.69	29.13	30.97	5.58	3.70	**0.82**	261.40	147.30
Darcy	Rel-ℓ_2	2.43×10^{-1}	1.71×10^{-1}	2.37×10^{-1}	3.95×10^{-1}	2.92×10^{0}	$\mathbf{6.58 \times 10^{-2}}$	1.15×10^{-1}	2.09×10^{-1}
	Rel-ℓ_1	1.27×10^{-1}	6.66×10^{-2}	1.23×10^{-1}	2.99×10^{-1}	2.72×10^{0}	$\mathbf{1.13 \times 10^{-2}}$	3.33×10^{-2}	3.87×10^{-2}
	Phys-Err	3.39×10^{-2}	1.15×10^{-2}	8.08×10^{-2}	1.24×10^{-1}	1.65×10^{1}	$\mathbf{1.05 \times 10^{-7}}$	9.45×10^{-2}	3.95×10^{-1}
	Time	36.55	29.51	31.49	5.85	3.98	**0.81**	257.90	154.20
NS	Rel-ℓ_2	4.30×10^{-1}	4.64×10^{-1}	3.68×10^{-1}	1.14×10^{0}	6.66×10^{1}	$\mathbf{2.22 \times 10^{-2}}$	3.17×10^{-3}	1.26×10^{-1}
	Rel-ℓ_1	2.57×10^{-1}	4.03×10^{-1}	2.52×10^{-1}	1.15×10^{0}	6.33×10^{1}	$\mathbf{1.76 \times 10^{-2}}$	2.15×10^{-3}	1.20×10^{-1}
	Phys-Err	9.83×10^{0}	1.06×10^{1}	1.39×10^{1}	2.62×10^{-3}	6.09×10^{5}	$\mathbf{8.04 \times 10^{-10}}$	2.50×10^{-4}	2.16×10^{0}
	Time	37.55	35.56	55.07	7.67	3.61	**1.20**	292.70	155.60

Table 1: Quantitative comparison on different PDEs, including relative ℓ_2 error (Rel-ℓ_2), relative ℓ_1 error (Rel-ℓ_1), physics error (Phys-Err), and inference time (s). We evaluate S^{3}GM and DiffPDE with a few-step sampling (50 steps) and a long-run setting (2000 steps), denoted as S^{3}GM-50 (2000) and DiffPDE-50 (2000). PerFlow is evaluated with 50 sampling steps. Best results are in bold and second best are underlined (2000-step baselines are excluded from ranking).

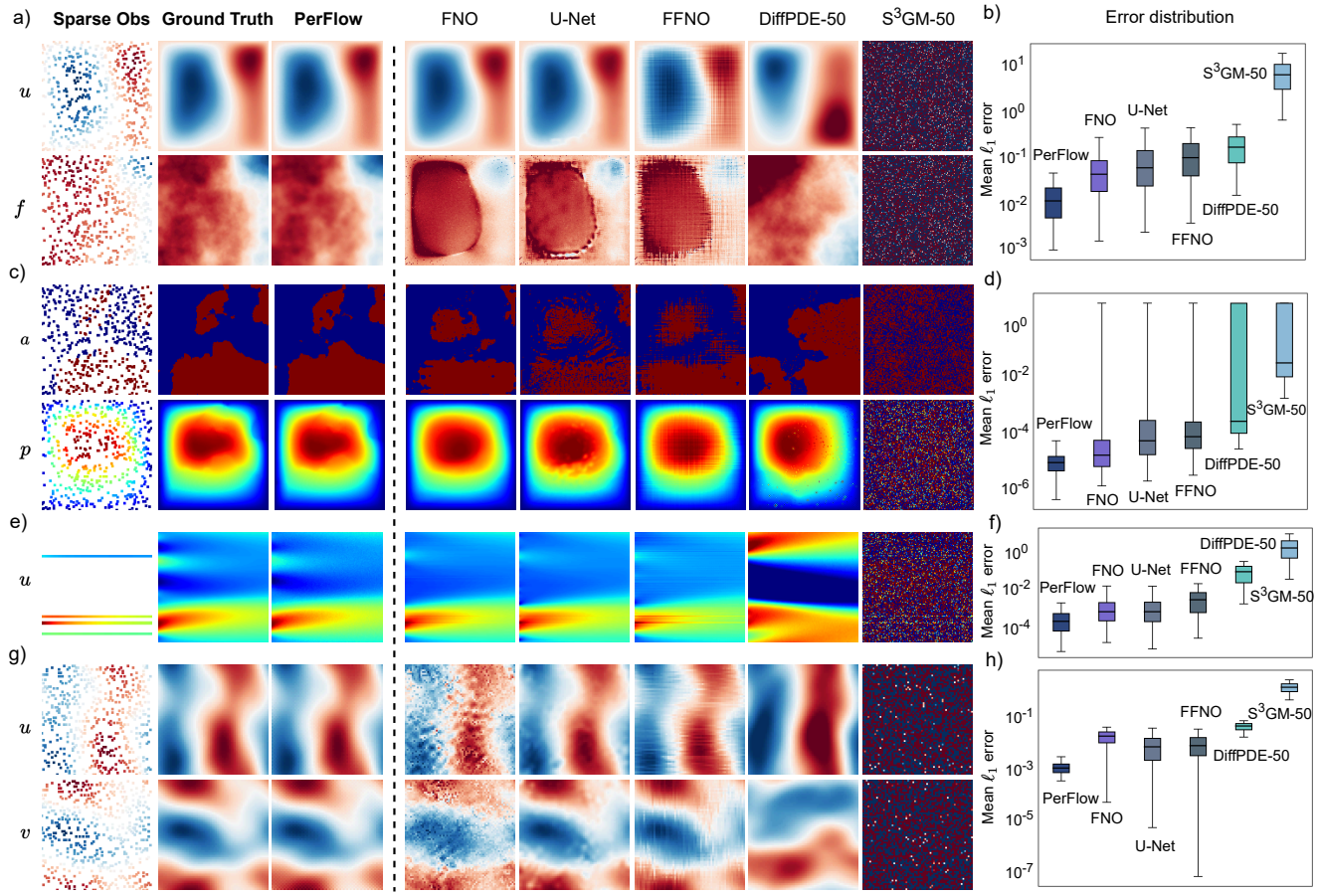

Figure 3: Sparse reconstruction physical fields (left) and the distribution of the point-wise mean ℓ_1 error (right) on four PDE benchmarks for each methods: **(a-b)** Poisson (solution and forcing), **(c-d)** Darcy (coefficient and pressure), **(e-f)** Burgers, and **(g-h)** Navier–Stokes equation (velocity components u and v).

5 Results

5.1 Benchmarks, Baselines, and Metrics

Benchmarks. We evaluate PerFlow on four kinds of PDE systems, including: (i) 1D *Burgers* with periodic boundary conditions, (ii) 2D *Poisson* with Dirichlet boundary conditions, (iii) 2D *Darcy* with Dirichlet boundary conditions, and (iv) 2D *incompressible Navier–Stokes* (NS) on a periodic domain. The default discretizations and measurements are as follows: Burgers uses 128 time steps on a 128-point spatial grid with $k = 5$ observed locations; Poisson and Darcy use 128×128 grids with $k = 500$ observed points; NS uses 2D velocity fields (u, v) on a 64×64 grid over $T = 10$ frames, with $k = 200$ observed locations per frame. Observation masks are randomly sampled. Additional dataset details are provided in Appendix 5 and Appendix 6.

Baselines. We compare PerFlow with (i) deterministic surrogates and (ii) guided generative baselines. For deterministic models, we train **FNO** [Li *et al.*, 2020], **U-Net** [Ronneberger *et al.*, 2015], and **FFNO** [Tran *et al.*, 2023] as *surrogate solvers*, learning the forward mapping between *full fields* governed by the PDE. For sparse reconstruction, we randomly initialize the full fields as input variables and iteratively optimize them by minimizing the mismatch between the model's output and the sparse observations. The reported inference time includes the entire optimization process. Furthermore, we also compare PerFlow with conditional FNO, which learns a deterministic mapping from sparse observations to the full field directly, and the corresponding results are reported in Appendix 8.

For generative baselines, we evaluate S^3GM [Li *et al.*, 2024a] and DiffPDE [Huang *et al.*, 2024], each with a few-step sampling (50 steps) and a long-run setting following the sampling steps used in the original papers (2000 steps), denoted as S^3GM-50 (2000) and DiffPDE-50 (2000), respectively. These two settings highlight the accuracy–efficiency trade-off: 2000 steps reflect the high-budget performance of guided baselines, while 50 steps compare them with PerFlow under a similar inference budget. Implementation details and baseline descriptions are provided in Appendix 3 and Appendix 4. **The code will be released at https://github.com/intell-sci-comput/PerFlow.**

Metrics. In this paper, we report reconstructed accuracy via relative errors, including Rel-$\ell_2 = \|\hat{\mathbf{x}} - \mathbf{x}\|_2 / \|\mathbf{x}\|_2$ and Rel-$\ell_1 = \|\hat{\mathbf{x}} - \mathbf{x}\|_1 / \|\mathbf{x}\|_1$. We also report the physics error (Phys-Err) for each PDE, such as divergence for NS and mass conservation for Burgers. Full definitions, including component averaging across variables and the PDE-specific Phys-Err computation, are provided in Appendix 7.

5.2 Sparse Reconstruction of Physical Fields

Fig. 3 shows the sparse reconstruction results for each PDE. Among these methods, deterministic models often offer oversmoothed reconstructed results and introduce artifacts in unobserved regions. Meanwhile, guided diffusion baselines, like DiffPDE and S^3GM, can recover several detail patterns and improve observation consistency when sufficient sampling steps are taken, but still struggle in the few-step regime.

Table 1 summarizes the quantitative results across 20 test cases for each PDE. PerFlow significantly outperforms baselines in reconstruction accuracy while strictly enforcing physical constraints. Compared to deterministic surrogates, PerFlow reduces Rel-ℓ_2 errors by *29.8%–94.0%* across the four benchmarks. This is because solving the inverse problem via iterative optimization with a deterministic model is ill-posed and sensitive to initialization, resulting in higher errors and prolonged inference times (29–55s). Moreover, compared with conditional FNO, PerFlow still achieves a remarkably high accuracy due to its ability to handle ill-posed scenarios (Appendix 8).

When compared with generative models under the same 50-step budget, PerFlow improves reconstruction accuracy by *1–2 orders of magnitude* over DiffPDE-50 and S^3GM-50. Simultaneously, PerFlow is *5.7–6.4*$\times$ faster than DiffPDE-50 and *3.0–4.2*$\times$ faster than S^3GM-50. This efficiency stems from PerFlow's design. Guided baselines require per-step backpropagation for observation and physics guidance, while PerFlow performs guidance-free sampling with lightweight projections. For example, on Burgers, mass projection introduces negligible overhead, with 0.79 ± 0.12s using projection versus 0.76 ± 0.06s without it. Even when the guided baselines are extended to 2000 sampling steps, PerFlow maintains superior accuracy on three of the four benchmarks while achieving significantly strong physics consistency. Notably, PerFlow provides up to a $\sim$320$\times$ speedup over DiffPDE-2000. In summary, PerFlow can not only offer high fidelity comparable to converged diffusion models, but also enjoy fast inference speed due to few-step solvers. Additional robustness analyses on base distributions and noisy observations are provided in the Appendix 9. Moreover, long-run guided baselines and varying-observation results are reported in Appendix 10 and Appendix 11.

5.3 Uncertainty Quantification

A main advantage of generative modeling is its ability to quantify posterior uncertainty under sparse observations. To verify this, we evaluate this capability on the Darcy and Burgers benchmarks, where the inverse problem is inherently ill-posed due to sparse and irregular measurements. Armed with the generative nature of PerFlow, we generate an ensemble of 20 independent predictions for each test case by sampling distinct initial noise states. We then calculate the predicted mean as the final reconstructed field and the entry-wise standard deviation across the ensemble to quantify the model's uncertainty. Fig. 4 shows the uncertainty quantification results. In all cases, the predictive mean accurately recovers the ground truth structures. Furthermore, we observe a strong spatial correlation between the estimated uncertainty and the actual reconstruction error. Regions exhibiting high variance typically coincide with larger deviations from the ground truth. For instance, uncertainty peaks at the sharp phase boundaries in the Darcy coefficient (Fig. 4**a**) and concentrates in the gaps between temporal snapshots for the Burgers equation (Fig. 4**c**). This alignment indicates that PerFlow effectively identifies challenging regions where the solution is less confident, serving as a reliable indicator of reconstruction errors.

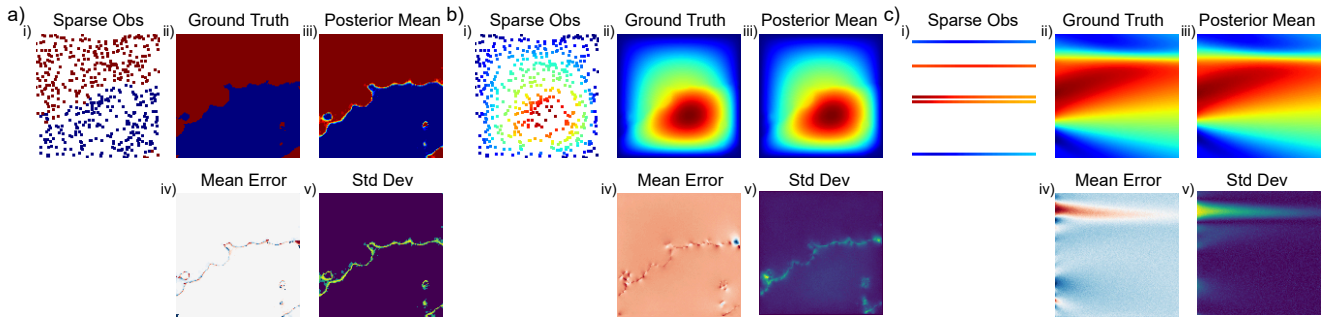

Figure 4: Uncertainty quantification under sparse observations, including Darcy coefficient $a(x)$ (**a**), Darcy pressure p (**b**), and 1D Burgers u (**c**). In each subfigure, panels (i)–(v) correspond to the sparse observations, the ground truth, the predicted mean value, the mean error, and the standard deviation over 20 samples.

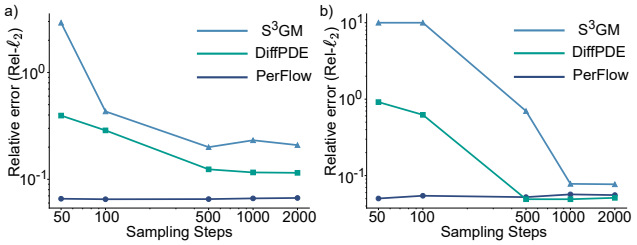

Figure 5: Reconstruction error versus sampling steps on (**a**) 2D Darcy and (**b**) 2D Poisson. We report the Rel-ℓ_2 error versus the number of sampling steps for S^{3}GM, DiffPDE, and PerFlow.

Method	Rel-ℓ_2	Rel-ℓ_1	Phys-Err
PerFlow	$\mathbf{8.63 \times 10^{-2}}$	$\underline{6.84 \times 10^{-2}}$	$\mathbf{4.47 \times 10^{-11}}$
RF-50	$\underline{8.79 \times 10^{-2}}$	7.16×10^{-2}	$\underline{7.09 \times 10^{-5}}$
FM-50	1.48×10^{-1}	1.40×10^{-1}	1.71×10^{-4}
DDIM-50	1.14×10^{-1}	9.92×10^{-2}	1.02×10^{-4}
DDPM-50	6.86×10^{0}	7.33×10^{0}	1.25×10^{-4}
DDPM-1000	9.10×10^{-2}	$\mathbf{6.46 \times 10^{-2}}$	1.10×10^{-4}

Table 2: Ablation studies across different generative samplers and flow variants on Burgers equation, including relative ℓ_2 error (Rel-ℓ_2), relative ℓ_1 error (Rel-ℓ_1), physics error (Phys-Err). Best results are in bold and second best are underlined.

5.4 Few-step Sampling

We study how reconstruction accuracy varies with the number of sampling steps for S^{3}GM, DiffPDE, and PerFlow (Fig. 5). Across these PDEs, PerFlow achieves strong accuracy in the few-step sampling regime and remains stable as the number of steps increases. In contrast, other guided diffusion methods exhibit a much slower and more severe trade-off between accuracy and the number of steps, because of the need to balance data-fidelity guidance and physics guidance. PerFlow addresses this issue via avoiding sampling-time backpropagation and embedding physics as hard constraints, thereby achieving efficient and accurate reconstruction.

5.5 Ablation Studies

We conduct ablations on 1D Burgers equation to study (i) the effect of *physics embedding* and (ii) the choice of *generative procedure*. Table 2 compares PerFlow with several variants, including *RF-50*, which removes the hard constraint projector from PerFlow; *FM-50*, which replaces rectified flow with a standard flow-matching model under the same 50-step sampling [Lipman *et al.*, 2022]; *DDIM-50* [Song *et al.*, 2020a] and *DDPM-50* [Ho *et al.*, 2020], two diffusion baselines sampled with 50 steps, and *DDPM-1000* with 1000 steps.

Comparing PerFlow with RF-50 reveals the contribution of hard constraint embedding. While both methods achieve similar reconstruction accuracy, PerFlow reduces physics error by several orders of magnitude (Table 2), indicating that constraint-preserving projection is crucial for producing physically consistent reconstructions. Furthermore, RF-based generation in PerFlow is more effective than other diffusion models with the same number of sampling (Table 2). DDPM-50 fails catastrophically, whereas DDIM-50 improves substantially but remains inferior to PerFlow in both accuracy and physical consistency. Although DDPM-1000 can achieve competitive Rel-ℓ_2/ℓ_1, it still exhibits larger physics error and requires an order-of-magnitude more sampling steps.

6 Conclusion and Future Work

We propose PerFlow, a physics-embedded rectified-flow framework for sparse reconstruction and uncertainty quantification in PDE-governed systems. Compared with posterior-guided generative methods, PerFlow decouples observation conditioning from physics enforcement by performing guidance-free observation conditioning in rectified-flow dynamics and encoding hard physics via constraint-preserving projection. Theoretically, sampling trajectories are guaranteed to remain on the physics-consistent manifold. Numerically, PerFlow achieves competitive accuracy and strong physics consistency across various PDE datasets, while enabling efficient conditional sampling (e.g., 50 steps) and up to $\sim$320$\times$ faster inference than 2000-step diffusion baselines.

There are two future works that we are interested in. Firstly, we can combine PerFlow with sensor placement approaches, which can maximize reconstruction quality and uncertainty reduction [Ma *et al.*, 2025]. Secondly, we can extend PerFlow to other downstream tasks, because fast conditional sampling on a physics-consistent manifold can provide reliable control and planning [Hu *et al.*, 2025a].

Acknowledgements

The work is supported by the Beijing Natural Science Foundation (No. F261002) and the National Natural Science Foundation of China (No. 62276269 and No. 62506367). R.Z. acknowledges support from the China Postdoctoral Science Foundation under Grant Number 2025M771582 and the Postdoctoral Fellowship Program of CPSF under Grant Number GZB20250408. The full version is available at http://arxiv.org/abs/2605.03548.

References

[Azizzadenesheli *et al.*, 2024] Kamyar Azizzadenesheli, Nikola Kovachki, Zongyi Li, Miguel Liu-Schiaffini, Jean Kossaifi, and Anima Anandkumar. Neural operators for accelerating scientific simulations and design. *Nature Reviews Physics*, 6(5):320–328, 2024.

[Bhaganagar and Chambers, 2025] Kiran Bhaganagar and David Chambers. Accelerated elliptical pde solver for computational fluid dynamics based on configurable u-net architecture: Analogy to v-cycle multigrid. *Machine Intelligence Research*, 22(2):324–336, 2025.

[Du *et al.*, 2024] Pan Du, Meet Hemant Parikh, Xiantao Fan, Xin-Yang Liu, and Jian-Xun Wang. Conditional neural field latent diffusion model for generating spatiotemporal turbulence. *Nature Communications*, 15(1):10416, 2024.

[Eliasof *et al.*, 2021] Moshe Eliasof, Eldad Haber, and Eran Treister. Pde-gcn: Novel architectures for graph neural networks motivated by partial differential equations. *Advances in neural information processing systems*, 34:3836–3849, 2021.

[Goodfellow *et al.*, 2020] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[He *et al.*, 2024] Junyan He, Seid Koric, Diab Abueidda, Ali Najafi, and Iwona Jasiuk. Geom-deeponet: A point-cloud-based deep operator network for field predictions on 3d parameterized geometries. *Computer Methods in Applied Mechanics and Engineering*, 429:117130, 2024.

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

[Holton and Hakim, 2013] James R Holton and Gregory J Hakim. *An introduction to dynamic meteorology*, volume 88. Academic press, 2013.

[Hu *et al.*, 2025a] Peiyan Hu, Xiaowei Qian, Wenhao Deng, Rui Wang, Haodong Feng, Ruiqi Feng, Tao Zhang, Long Wei, Yue Wang, Zhi-Ming Ma, and Tailin Wu. From uncertain to safe: Conformal adaptation of diffusion models for safe PDE control. In *Forty-second International Conference on Machine Learning*, 2025.

[Hu *et al.*, 2025b] Peiyan Hu, Rui Wang, Xiang Zheng, Tao Zhang, Haodong Feng, Ruiqi Feng, Long Wei, Yue Wang, Zhi-Ming Ma, and Tailin Wu. Wavelet diffusion neural operator. In *The Thirteenth International Conference on Learning Representations*, 2025.

[Huang *et al.*, 2024] Jiahe Huang, Guandao Yang, Zichen Wang, and Jeong Joon Park. Diffusionpde: Generative pde-solving under partial observation. *Advances in Neural Information Processing Systems*, 37:130291–130323, 2024.

[Kingma and Welling, 2013] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[Lapidus and Pinder, 1999] Leon Lapidus and George F Pinder. *Numerical solution of partial differential equations in science and engineering*. John Wiley & Sons, 1999.

[Li *et al.*, 2020] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020.

[Li *et al.*, 2023] Zijie Li, Dule Shu, and Amir Barati Farimani. Scalable transformer for pde surrogate modeling. *Advances in Neural Information Processing Systems*, 36:28010–28039, 2023.

[Li *et al.*, 2024a] Zeyu Li, Wang Han, Yue Zhang, Qingfei Fu, Jingxuan Li, Lizi Qin, Ruoyu Dong, Hao Sun, Yue Deng, and Lijun Yang. Learning spatiotemporal dynamics with a pretrained generative model. *Nature Machine Intelligence*, 6(12):1566–1579, 2024.

[Li *et al.*, 2024b] Zongyi Li, Hongkai Zheng, Nikola Kovachki, David Jin, Haoxuan Chen, Burigede Liu, Kamyar Azizzadenesheli, and Anima Anandkumar. Physics-informed neural operator for learning partial differential equations. *ACM/IMS Journal of Data Science*, 1(3):1–27, 2024.

[Li *et al.*, 2025] Zijie Li, Anthony Zhou, and Amir Barati Farimani. Generative latent neural pde solver using flow matching. *arXiv preprint arXiv:2503.22600*, 2025.

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

[Lippe *et al.*, 2023] Phillip Lippe, Bas Veeling, Paris Perdikaris, Richard Turner, and Johannes Brandstetter. Pde-refiner: Achieving accurate long rollouts with neural pde solvers. *Advances in Neural Information Processing Systems*, 36:67398–67433, 2023.

[Liu *et al.*, 2022] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

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

[Lu *et al.*, 2025] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *Machine Intelligence Research*, 22(4):730–751, 2025.

[Ma *et al.*, 2025] Yuezhou Ma, Haixu Wu, Hang Zhou, Huikun Weng, Jianmin Wang, and Mingsheng Long. Physense: Sensor placement optimization for accurate physics sensing. *arXiv preprint arXiv:2505.18190*, 2025.

[Rahman *et al.*, 2022] Md Ashiqur Rahman, Zachary E Ross, and Kamyar Azizzadenesheli. U-no: U-shaped neural operators. *arXiv preprint arXiv:2204.11127*, 2022.

[Rezende and Mohamed, 2015] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

[Ronneberger *et al.*, 2015] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[Shysheya *et al.*, 2024] Aliaksandra Shysheya, Cristiana Diaconu, Federico Bergamin, Paris Perdikaris, José Miguel Hernández-Lobato, Richard Turner, and Emile Mathieu. On conditional diffusion models for pde simulations. *Advances in Neural Information Processing Systems*, 37:23246–23300, 2024.

[Song *et al.*, 2020a] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[Song *et al.*, 2020b] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[Strogatz, 2024] Steven H Strogatz. *Nonlinear dynamics and chaos: with applications to physics, biology, chemistry, and engineering*. Chapman and Hall/CRC, 2024.

[Tadmor, 2012] Eitan Tadmor. A review of numerical methods for nonlinear partial differential equations. *Bulletin of the American Mathematical Society*, 49(4):507–554, 2012.

[Tran *et al.*, 2023] Alasdair Tran, Alexander Mathews, Lexing Xie, and Cheng Soon Ong. Factorized fourier neural operators. In *The Eleventh International Conference on Learning Representations*, 2023.

[Tu *et al.*, 2023] Jiyuan Tu, Guan Heng Yeoh, Chaoqun Liu, and Yao Tao. *Computational fluid dynamics: a practical approach*. Elsevier, 2023.

[Wan *et al.*, 2025] Han Wan, Rui Zhang, Qi Wang, Yang Liu, and Hao Sun. Pesanet: Physics-encoded spectral attention network for simulating pde-governed complex systems. In James Kwok, editor, *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pages 7751–7759. International Joint Conferences on Artificial Intelligence Organization, 8 2025. Main Track.

[Wang *et al.*, 2025] Sifan Wang, Zehao Dou, Siming Shan, Tong-Rui Liu, and Lu Lu. Fundiff: Diffusion models over function spaces for physics-informed generative modeling. *arXiv preprint arXiv:2506.07902*, 2025.

[Wei *et al.*, 2024] Long Wei, Peiyan Hu, Ruiqi Feng, Haodong Feng, Yixuan Du, Tao Zhang, Rui Wang, Yue Wang, Zhi-Ming Ma, and Tailin Wu. Diffphycon: A generative approach to control complex physical systems. *Advances in Neural Information Processing Systems*, 37:4090–4147, 2024.

[Wu *et al.*, 2024] Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for pdes on general geometries. *arXiv preprint arXiv:2402.02366*, 2024.

[Xiong *et al.*, 2024] Wei Xiong, Xiaomeng Huang, Ziyang Zhang, Ruixuan Deng, Pei Sun, and Yang Tian. Koopman neural operator as a mesh-free solver of non-linear partial differential equations. *Journal of Computational Physics*, 513:113194, 2024.

[Xu *et al.*, 2022] Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. Geodiff: A geometric diffusion model for molecular conformation generation. *arXiv preprint arXiv:2203.02923*, 2022.

[Zeng *et al.*, 2024] Bocheng Zeng, Qi Wang, Mengtao Yan, Yang Liu, Ruizhi Chengze, Yi Zhang, Hongsheng Liu, Zidong Wang, and Hao Sun. Phympgn: Physics-encoded message passing graph network for spatiotemporal pde systems. *arXiv preprint arXiv:2410.01337*, 2024.

[Zhang *et al.*, 2015] Zheng Zhang, Yong Xu, Jian Yang, Xuelong Li, and David Zhang. A survey of sparse representation: algorithms and applications. *IEEE access*, 3:490–530, 2015.

[Zhang *et al.*, 2024] Rui Zhang, Qi Meng, and Zhi-Ming Ma. Deciphering and integrating invariants for neural operator learning with various physical mechanisms. *National Science Review*, 11(4):nwad336, 2024.

[Zhang *et al.*, 2025] Rui Zhang, Qi Meng, Rongchan Zhu, Yue Wang, Wenlei Shi, Shihua Zhang, Zhi-Ming Ma, and Tie-Yan Liu. Monte carlo neural pde solver for learning pdes via probabilistic representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.