DiffusionRollout: Uncertainty-Aware Rollout Planning in Long-Horizon PDE Solving

Anonymous authors
Paper under double-blind review

Abstract

We propose DiffusionRollout, a novel selective rollout planning strategy for autoregressive diffusion models, aimed at mitigating error accumulation in long-horizon predictions of physical systems governed by partial differential equations (PDEs). Building on the recently validated probabilistic approach to PDE solving, we further explore its ability to quantify predictive uncertainty and demonstrate a strong correlation between prediction errors and standard deviations computed over multiple samples—supporting their use as a proxy for the model's predictive confidence. Based on this observation, we introduce a mechanism that adaptively selects step sizes during autoregressive rollouts, improving long-term prediction reliability by reducing the compounding effect of conditioning on inaccurate prior outputs. Extensive evaluation on long-trajectory PDE prediction benchmarks validates the effectiveness of the proposed uncertainty measure and adaptive planning strategy, as evidenced by lower prediction errors and longer predicted trajectories that retain a high correlation with their ground truths.

1 Introduction

Solving partial differential equations (PDEs) is fundamental to scientific computing, enabling the modeling and prediction of a wide range of physical systems—from microscopic phenomena like pattern formation driven by chemical reactions and diffusion, to macroscopic dynamics such as ocean currents that influence global climate. While traditional approaches have relied on numerical solvers due to the lack of closed-form solutions, recent advances such as physics-informed neural networks (PINNs) (Raissi et al., 2019; Krishnapriyan et al., 2021) and neural operators (Li et al., 2020; 2021) have opened new possibilities for reducing computational cost and accelerating solution times. Besides, learning-based solvers enable data-driven simulations of real-world systems with unknown governing equations, as they can be trained directly from observational data without requiring prior knowledge of the underlying dynamics. The computational efficiency and flexibility of learning-based PDE solvers have led to their success across diverse domains, including fluid dynamics (Li et al., 2023c; 2024; Wu et al., 2024) and climate modeling (Kurth et al., 2023).

The standard approach to training neural PDE solvers casts the problem as regression, minimizing empirical risk between model predictions and ground truth. Recent work (Huang et al., 2024; Shysheya et al., 2024), however, adopt a generative perspective, framing PDE modeling as conditional generation—sampling plausible future trajectories conditioned on past observations. This shift has enabled neural solvers to tackle more complex tasks, such as inverse problems (Huang et al., 2024) and data assimilation (Shysheya et al., 2024), that conventional regression-based methods struggle to address. Moreover, even in purely regression-based settings, the training objective, originally designed for modeling high-dimensional data distributions in diffusion and score-based generative models (Song & Ermon, 2019; Ho et al., 2020; Song et al., 2021), naturally extends to PDE learning, where discretized solution functions are typically high-dimensional (Hu et al., 2025). More importantly, it mitigates spectral bias—where regression losses overemphasize low-frequency components (Lippe et al., 2023)—which adversely affects the long-term stability of solvers, particularly in nonlinear systems.

In this work, we further explore benefits of generative modeling for PDEs, focusing on the challenging task of long-term prediction through autoregressive rollouts, where the model recursively uses its own past predictions to generate future timesteps. Although this approach has been empirically shown to outperform direct prediction methods (Wang & Perdikaris, 2023; Li et al., 2022)—which parameterize the solver to estimate the system state at a target time from initial and boundary conditions—it also suffers from a key limitation: exposure bias (Arora et al., 2022), where discrepancies between the model's predictions and the training data accumulate over time, leading to compounding errors. This issue becomes particularly pronounced in long-term predictions of physical systems described by continuous functions over space and time, where even small prediction errors can rapidly accumulate and destabilize the solution.

The core idea behind DiffusionRollout that addresses this challenge is to take advantage of the probabilistic nature of diffusion models and incorporate sampling-based uncertainty estimates. Our analysis reveals that statistics computed from multiple samples drawn from a conditional diffusion model—trained to approximate the distribution of possible future trajectories—are strongly correlated with prediction errors. This makes them a reliable metric for estimating the model's confidence in its predictions during rollouts. Based on this, we propose a simple yet effective training-free selective planning strategy, where unreliable future predictions are discarded and the model advances only to timesteps where it is sufficiently confident. In experiments, we task neural PDE solvers with challenging long-trajectory predictions and demonstrate the effectiveness of the proposed framework, which sets a new state-of-the-art over recent neural operators. Moreover, comparisons with the base model and the original rollout strategy—where the sliding window advances by the largest steps possible—clearly show performance gains achieved without any additional training.

2 Related Work

Neural PDE Solvers and Long-Term Rollouts. Neural PDE solvers offer several advantages over traditional numerical methods, including modeling of physical systems with unknown governing equations using physics-informed neural networks (PINNs) (Raissi et al., 2019; Krishnapriyan et al., 2021; Cho et al., 2024) or neural operators (Li et al., 2020; 2021), prediction of system dynamics from partial or incomplete observations (Huang et al., 2024), and solving control tasks (Hu et al., 2025). Together with the development of network architectures specialized in surrogate learning (Li et al., 2023c; 2024; 2023b;a; Hao et al., 2023; Xiao et al., 2024; Kitaev et al., 2020; Choromanski et al., 2021; Cao, 2021; Wu et al., 2024; Wang & Wang, 2024; Alkin et al., 2024), there is increasing interest in applying such models to achieve accurate long-term rollouts, often spanning hundreds of timesteps (Lippe et al., 2023; Ruhe et al., 2024; Shysheya et al., 2024). Lippe et al. (2023) adapt the iterative denoising scheme of diffusion models (Song & Ermon, 2019; Ho et al., 2020; Song et al., 2021) to better capture high-frequency components in predicted solutions, which is especially critical for nonlinear PDEs. Similarly, Shysheya et al. (2024) investigate conditioning strategies for diffusion models in PDE learning and introduce an autoregressive sampling scheme for long-horizon predictions. Likewise, Ruhe et al. (2024) propose an alternative diffusion time scheduling method that better captures the varying uncertainty across the prediction horizon. Building on this line of work, we propose a simple yet effective approach to quantify predictive uncertainty and leverage it to further reduce error during long rollouts.

Exposure Bias in Autoregressive Generation. Exposure bias (Ranzato et al., 2016; Bengio et al., 2015; Arora et al., 2022) refers to the discrepancy between training and inference in autoregressive models: during training, the model is conditioned on ground-truth sequences, whereas at inference time, it must rely on its own previous predictions. This mismatch can lead to error accumulation, particularly in long-horizon generation tasks such as language modeling (OpenAI, 2024), visual autoregressive generation (Tian et al., 2024; Han et al., 2024), and PDE solving (Li et al., 2023b; 2021; Wu et al., 2024; Hu et al., 2025; Lippe et al., 2023; Shysheya et al., 2024). To address this issue, a variety of training strategies have been proposed, including adversarial training (Xu et al., 2019), reinforcement learning (Bahdanau et al., 2017; Chen et al., 2020), scheduled sampling or teacher-forcing alternatives (Zhang et al., 2019; Leblond et al., 2018), and unlikelihood-based objectives (Welleck et al., 2020). Despite these advances, most prior work focuses on modifying the training procedure, often requiring models to be retrained from scratch. In contrast, we propose a simple and practical approach to mitigating exposure bias in long-horizon PDE solving without

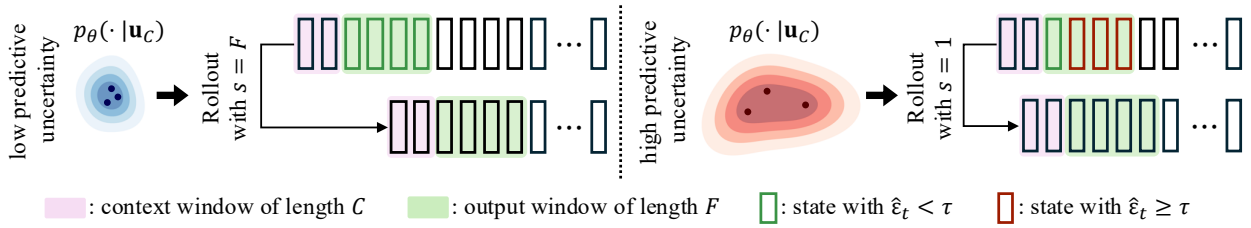

Figure 1: **Adaptive rollout step size s based on predictive uncertainty.** The figure illustrates how predictive uncertainty $\hat{\varepsilon}_t$ governs the rollout step size. On the left, low sample variance leads all future states to satisfy $\hat{\varepsilon}_t < \tau$, allowing a full-step rollout ($s = F$). On the right, high variance (i.e., uncertainty) causes only the first state to pass the threshold, leading to $s = 1$. Based on this, our diffusion-based PDE solver adaptively adjusts the rollout step size during autoregressive prediction.

retraining. Our method adaptively adjusts the rollout step size based on the model's prediction uncertainty, balancing a trade-off between network approximation error and inaccurate condition-induced error.

Uncertainty Quantification Using Diffusion Models. Uncertainty quantification in learning-based models has been extensively studied as a means of estimating the confidence of model predictions; we refer the reader to the comprehensive survey (Gawlikowski et al., 2023) and provide a brief summary below. Common approaches for estimating uncertainty include aggregating predictions from an ensemble of models (Lakshminarayanan et al., 2017), or training Bayesian neural networks (BNNs) (Depeweg et al., 2018) that model posterior distributions over network weights and make predictions by marginalizing over the learned prior. Similar techniques have only recently been applied to diffusion models (Berry et al., 2024; Chan et al., 2024). Berry et al. (2024) estimate epistemic uncertainty by training multiple submodules of a pre-trained diffusion model and computing mutual information between outputs and weights, whereas Chan et al. (2024) use Bayesian hyper-networks to generate ensembles of model parameters to bypass the need for training multiple networks. However, these approaches inherit the limitations of ensembles and Bayesian neural networks, including high training costs and the need for specialized network designs. In contrast, our method is architecture-agnostic and requires no additional training.

3 Long-Horizon Rollouts of PDE Solutions

3.1 Learning to Solve Time-Dependent PDEs

We focus on time-dependent partial differential equations (PDEs) defined over a temporal domain $t \in [0, T]$ and a d-dimensional spatial domain $\Omega \subseteq \mathbb{R}^d$. In general, the solution $\mathbf{u}(\mathbf{x}, t) : \Omega \times [0, T] \to \mathbb{R}^c$, with temporal and spatial derivatives satisfies:

$$\frac{\partial \mathbf{u}}{\partial t} = F(\mathbf{u}, \frac{\partial \mathbf{u}}{\partial \mathbf{x}}, \frac{\partial^2 \mathbf{u}}{\partial \mathbf{x}^2}, \cdots) + \mathbf{f}(\mathbf{x}, t), \tag{1}$$

where F denotes a differential operator and $\mathbf{f}(\mathbf{x}, t) : \Omega \times [0, T] \to \mathbb{R}^c$ is a forcing function. Typically, Eqn. 1 is accompanied by the initial condition $\mathbf{u}(\mathbf{x}, 0) = \mathbf{u}_0(\mathbf{x})$ and the boundary condition $B[\mathbf{u}](\mathbf{x}, t) = 0$ for $(\mathbf{x}, t) \in \partial\Omega \times [0, T]$. The main goal of neural surrogate learning is to approximate the solution operator $\mathcal{G} : \mathcal{U} \to \mathcal{U}$ for Eqn. 1 using a neural network $\mathcal{G}_\theta$, parameterized by θ.

Let $\mathbf{u}_{0:T} := [\mathbf{u}(\mathbf{x}, 0), \ldots, \mathbf{u}(\mathbf{x}, T)] \in \mathbb{R}^{(T+1)\times N \times c}$ denote the discretized solution trajectory, obtained by sampling $\mathbf{u}(\mathbf{x}, t)$ at N spatial points in the domain Ω over T timesteps, each separated by $\Delta t \in \mathbb{R}_{\geq 0}$. When T is small, the entire trajectory can be modeled from the initial condition $\mathbf{u}_0$ using a single forward pass of the model. However, this direct entire trajectory modeling approach has been shown to fall short for large T—where T spans tens to hundreds of steps and each temporal solution slice $\mathbf{u}_t$ is a high-dimensional vector. Instead, autoregressive rollouts have proven more effective than modeling the entire trajectory directly at large T (Lippe et al., 2023; Li et al., 2022; Wang & Perdikaris, 2023).

As such, we choose to model short segments of length $T' \ll T$, where each segment is divided into a context of length C and a future portion of length $F = T' - C$, serving as the input and target output of $\mathcal{G}_\theta$, respectively. At inference time, the model $\mathcal{G}_\theta$ predicts long trajectories in an autoregressive manner, using its previous outputs as inputs for the next segment. For training, an entire trajectory $\mathbf{u}_{0:T}$ is now chunked into M pairs of a context and a future portion $\{(\mathbf{u}_{C_i}, \mathbf{u}_{F_i})\}_{i=1}^{M}$, with $C_i = \{t_i, t_i + 1, \ldots, t_i + C - 1\}$ and $F_i = \{t_i + C, t_i + C + 1, \ldots t_i + T' - 1\}$ indicating the sets of timestep indices belonging to the context and the future portions of the i-th pair. Note that we use C and F interchangeably to denote both the index sets and their corresponding cardinalities, for notational convenience. In this setting, the parameters θ are optimized using these pairs by minimizing the empirical risk:

$$\mathcal{L} = \mathbb{E}_{\mathbf{u}_{0:T} \sim \mathcal{D}} \left[\sum_{i=1}^{M} \| \mathbf{u}_{F_i} - \mathcal{G}_\theta(\mathbf{u}_{C_i}) \|^2 \right], \tag{2}$$

where $\mathcal{D}$ is the training set. Once trained, the model $\mathcal{G}_\theta$ serves as an approximator of the underlying mapping from context to future, i.e., $\mathbf{u}_F \approx \mathcal{G}_\theta(\mathbf{u}_C)$.

3.2 Generative Models in PDE Solving

While regression models trained with Eqn. 2 are commonly used in PDE modeling, recent studies have demonstrated the effectiveness of probabilistic approaches to learning $\mathcal{G}_\theta$. These methods not only enable applications where generative capabilities are essential (e.g., inverse problems) (Huang et al., 2024; Shysheya et al., 2024), but also improve performance in purely predictive tasks (Lippe et al., 2023; Hu et al., 2025). From the perspective of generative modeling, particularly with diffusion or score-based models (Ho et al., 2020; Song et al., 2021), the mapping $\mathbf{u}_F = \mathcal{G}_\theta(\mathbf{C})$ can be viewed as sampling from a conditional distribution. In this framework, a random sample $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is gradually transformed into the target $\mathbf{u}_F$—the future portion of a segment—that is plausible given the context $\mathbf{u}_C$:

$$\mathbf{u}_F \sim p_\theta(\mathbf{u}_F | \mathbf{u}_C) \tag{3}$$

where $p_\theta(\mathbf{u}_F|\mathbf{u}_C)$ is a conditional distribution of possible $\mathbf{u}_F$'s. Given sufficient past context $\mathbf{u}_C$, $p_\theta(\mathbf{u}_F|\mathbf{u}_C)$ collapses to a narrow region encompassing a set of plausible future states $\mathbf{u}_F$ with only minor variations. These can be sampled through an iterative denoising or refinement process, where $\mathbf{u}_F$ is progressively reconstructed by recovering both low- and high-frequency components while removing noises of decreasing magnitudes from the initial sample $\boldsymbol{\epsilon}$. Compared to neural solvers trained with regression-based losses, such as the one illustrated in Eqn. 2, this approach has been shown to better capture a broader range of frequencies in the target signal $\mathbf{u}_F$—a key factor in stabilizing long rollouts for nonlinear systems (Lippe et al., 2023). In the following section, we investigate how this probabilistic interpretation naturally enables the quantification of the solver's prediction uncertainty or confidence. We then propose a planning strategy that improves prediction accuracy through a minimal modification to the rollout procedure, without requiring specialized training strategies or changes to the model architecture.

3.3 Adaptive Rollout Step Size via Uncertainty Estimation

When adopting the C-in, F-out autoregressive scheme discussed in Sec. 3.1, addressing error accumulation is critical for ensuring long-horizon stability. We identify two primary sources of such accumulation: (1) **network approximation error**, which arises from the inherent limitations of the learned model $\mathcal{G}_\theta$ in approximating the true solution operator and tends to increase with the number of network function evaluations (NFEs); and (2) **condition-induced error**, which stems from using inaccurate or uncertain predicted states as context in subsequent steps. These two sources are closely interrelated: a prediction error caused by approximation inaccuracies is likely to propagate as a condition-induced error in subsequent steps, compounding over time and destabilizing the entire trajectory (Ranzato et al., 2016; Bengio et al., 2015; Arora et al., 2022).

Empirically, we observe that across the model output $\mathbf{u}_F$, the prediction error tends to increase with temporal distance from the context $\mathbf{u}_C$. We attribute this trend not only to the inherent difficulty of modeling long-range dependencies, but also to the fact that temporally adjacent frames are more frequently included

together in training segments, following the training procedure described in Sec. 3.1. This observation implies that F predictions within the same time window covered by $\mathcal{G}_\theta$ should *not* be trusted equally, and that overall model performance can be improved by selectively using only the more reliable predictions as context for subsequent steps.

Specifically, let $s \in \{1, 2, \ldots, F\}$ denote the step size for selecting predictions from $\mathbf{u}_F$. A straightforward strategy to mitigate condition-induced error is to follow the most conservative policy by setting $s = 1$, accepting only a single prediction from $\mathbf{u}_F$, discarding the remaining $F-1$ predictions, and advancing the time window by just one step. While this reduces reliance on potentially inaccurate predictions, it leads to a higher number of network function evaluations (NFEs), thereby increasing the likelihood of accumulating network approximation error. On the other hand, using the largest step size $s = F$ accepts all F predictions and uses the last C of them as context for the next rollout. This reduces the number of NFEs and, consequently, the approximation error. However, it increases reliance on less accurate predictions, which amplifies condition-induced error.

Therefore, an ideal way to suppress error accumulation is to identify which type of error—network approximation error or condition-induced error—is more dominant and adapt the rollout step size accordingly. Inspired by adaptive step size methods in numerical analysis (Runge, 1895; Kutta, 1901; Hairer et al., 1987), we propose a simple yet effective adaptive rollout step size strategy.

However, the true prediction error required to determine the rollout step size is inaccessible at inference time. While conventional adaptive step size methods address this limitation by estimating error through high-order approximations, such estimates are often computationally expensive and sensitive to numerical instability. To this end, we propose an surrogate measure that estimates model confidence and enables the rejection of unreliable outputs during autoregerssive rollouts.

Compared to deterministic regression models, a key advantage of modeling a PDE solver as a generative model is its ability to estimate uncertainty through sampling. Predictions for uncertain future timesteps tend to vary across different samples drawn from $p_\theta(\mathbf{u}_F \mid \mathbf{u}_C)$, providing a natural measure of uncertainty. We leverage this by computing statistical measures—specifically, standard deviation—over these samples as a surrogate for the model's predictive uncertainty.

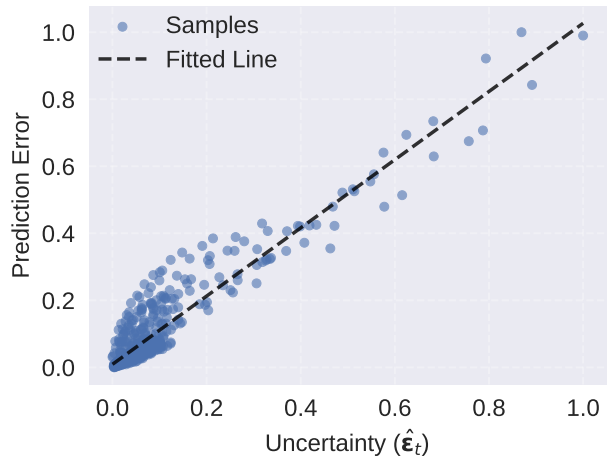

Figure 2: **Analysis of predictive uncertainty $\hat{\varepsilon}_t$ as a surrogate for prediction error.** The proposed quantity exhibits a strong correlation with the true prediction error, as indicated by the correlation coefficient close to 1 ($r = 0.94$).

Formally, for each $t \in F$, the predicted state $\mathbf{u}_t \in \mathbb{R}^{N \times c}$ refers to the t-th slice of the future states $\mathbf{u}_F \in \mathbb{R}^{F \times N \times c}$. The uncertainty $\hat{\boldsymbol{\varepsilon}}_t$ of $\mathbf{u}_t$ is defined as the norm of the standard deviations of the field elements, computed from a set of samples $\{\mathbf{u}_t^{(k)}\}_{k=1}^K$, where K is the number of samples drawn from the conditional distribution $p_\theta(\cdot \mid \mathbf{u}_C)$:

$$\hat{\boldsymbol{\varepsilon}}_t = \|\sigma\left(\{\mathbf{u}_t^{(k)}\}_{k=1}^K\right)\|_2. \tag{4}$$

As evidenced by the right inset, the proposed uncertainty measure $\hat{\boldsymbol{\varepsilon}}_t$ shows a strong correlation with the actual prediction error in the analysis conducted on one of the benchmark datasets (Gray-Scott Reaction-Diffusion Equation) discussed in Sec. 4. This supports the use of uncertainty as a surrogate for prediction error. Leveraging this, at each prediction step, we compare the uncertainty of each predicted state to a predefined tolerance τ, and accept only the states whose uncertainty falls below τ for use as context in the subsequent rollout. The rollout step size s is then selected as :

$$s = \max(\{t \in \{1, 2, \ldots, F\} | \hat{\boldsymbol{\varepsilon}}_t < \tau\} \cup \{1\}). \tag{5}$$

We determine τ used in our experiments through binary search on a subset of the training set, where ground-truth prediction errors are available. While more sophisticated, learning-based approaches could be explored to predict optimal τ values, we find that this simple strategy already yields noticeable performance improvements over the previously used sampling method.

4 Experiments

4.1 Experiment Setup

We evaluate the effectiveness of our method across various PDE types through both qualitative and quantitative comparisons against state-of-the-art neural PDE solvers.

Baselines. We compare DiffusionRollout against state-of-the-art neural operators, including regression-based methods—FNO (Li et al., 2021), FactFormer (Li et al., 2023b), and Transolver (Wu et al., 2024)—as well as the diffusion-based neural operators—PDE-Refiner (Lippe et al., 2023) and WDNO (Hu et al., 2025). For a fair comparison, all models are trained to predict six future states from four past observations with the parameter count standardized to approximately 3 million except for WDNO (Hu et al., 2025), which uses 21 million parameters due to convergence issue.

Datasets. We benchmark on challenging *long*-trajectory simulations of length $T \approx 100$, where autoregressive rollout is crucial for inference. Unless otherwise noted, all simulations are conducted on a unit square discretized at a 64×64 resolution. Our evaluation spans diverse PDE systems—ranging from reaction-diffusion-driven pattern formations to turbulent flow, as detailed below. Across all PDE systems, we assume the underlying governing equations are *unknown* and incorporate no prior knowledge during training or inference.

1. **Gray-Scott Reaction-Diffusion Equation**: Gray-Scott Reaction-Diffusion models the evolution of the concentrations $X(\mathbf{x}, t)$ and $Y(\mathbf{x}, t)$ of two chemical species, undergoing simultaneous reaction and diffusion:

$$\frac{\partial X}{\partial t} = \delta_X \Delta X - XY^2 + f(1 - X), \quad \frac{\partial Y}{\partial t} = \delta_Y \Delta Y - XY^2 - (f + k)Y,$$

   where δ_X and δ_Y are the diffusion coefficients, and f and k denote the feed and kill rates, respectively. The concentrations $X(\mathbf{x}, t)$ and $Y(\mathbf{x}, t)$, evolving over space and time, produce complex patterns that are highly sensitive to the parameters f and k. This sensitivity makes the problem an ideal testbed, as diverse and intricate trajectories can be generated by varying these values. We use 1,200 trajectories from `The Well` dataset (Ohana et al., 2024), following the provided train-test split and subsampling every 10 frames to obtain sequences of length 101. At test time, all models are tasked with predicting 97 future states based on observations from the first four timesteps of each simulation.

2. **2D Turbulent Radiative Flow Equation**: In this setup, we consider a turbulent flow simulation as a benchmark problem, which models turbulent interaction between two gases—one cold and one hot—initially moving relative to each other. As they gradually mix over time, they form a highly reactive intermediate-temperature gas. This behavior of the system is modeled by the equation:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho v) = 0, \quad \frac{\partial \rho v}{\partial t} + \nabla \cdot (\rho v v + P) = 0,$$
$$\frac{\partial E}{\partial t} + \nabla \cdot ((E + P)v) = -\frac{E}{t_{\text{cool}}}, \quad E = \frac{P}{\gamma - 1},$$

   where $\gamma = \frac{5}{3}$, with ρ denoting the density, v the velocity, P the pressure, E the total energy, and t_{cool} the cooling time. Due to the highly stochastic nature of gas mixing, accurately predicting future states during rollouts becomes challenging, making this setup suitable for our experiments. As with the Gray-Scott reaction-diffusion system, we use 90 trajectories of length 101 from the `The Well` dataset (Ohana et al., 2024; Fielding et al., 2020), following the same training and test splits. All models use the first seven timesteps as context and are tasked with predicting the remaining 94 future states.

3. **Cahn-Hilliard Equation**: Cahn–Hilliard equation models phase separation in binary mixtures, capturing how two components—such as oil and water—gradually separate into distinct regions over time. Letting $c(\mathbf{x}, t)$ denote the concentration of one component, the equation is given by:

$$\frac{\partial c}{\partial t} = M\left(-\kappa \nabla^4 c + \nabla^2 f'(c)\right),$$

where M denotes the mobility, and $f(c) = Wc^2(1-c)^2$ is the bulk free energy density, with W representing the height of the thermodynamic barrier. The trajectories exhibit continuously evolving component concentrations, with dynamic patterns that emerge and fade over time. Using a numerical solver (Soares et al., 2023), we generate 960 training trajectories and 120 test trajectories, each consisting of simulation results over 101 timesteps. During inference, models are given the first seven timesteps as context and tasked with predicting the remaining sequence.

4. **Anisotropic Diffusion**: The final problem setup we consider is anisotropic diffusion, a generalized form of the heat equation in which the diffusion coefficient varies across the domain:

$$\frac{\partial u}{\partial t} = \nabla \cdot A \nabla u, \tag{6}$$

where u denotes the evolving scalar field (e.g., temperature or intensity), and A is a spatially varying, typically positive-definite diffusion tensor that controls the local direction and rate of diffusion. To generate data, we use the anisotropic diffusion simulator from APEBench (Koehler et al., 2024), producing a total of 1080 trajectories, which are split into 960 for training and 120 for testing. All simulations are run with the default parameters provided by the benchmark. At inference time, the models predict the remaining 97 frames using the first four frames as context.

Evaluation Metrics. Given a ground-truth solution $\hat{\mathbf{u}}$ and a model's corresponding prediction $\mathbf{u}$, we use the relative $\mathcal{L}_2$ error to evaluate the accuracy of model predictions:

$$\text{Rel } \mathcal{L}_2(\mathbf{u}, \hat{\mathbf{u}}) = \frac{\|\mathbf{u} - \hat{\mathbf{u}}\|_2}{\|\hat{\mathbf{u}}\|_2}. \tag{7}$$

Furthermore, following PDE-Refiner (Lippe et al., 2023), to evaluate long-term stability, we additionally compute the per-timestep Pearson correlation coefficeint $r(t)$ between the predicted and ground-truth states at each timestep. Specifically, for each time index t and , we compute Pearson correlation $r(t)$ between the predicted state $\mathbf{u}_t$ and ground-truth state $\hat{\mathbf{u}}_t$ as:

$$r(t) = \frac{\text{Cov}(\mathbf{u}_t, \hat{\mathbf{u}}_t)}{\sqrt{\text{Var}(\mathbf{u}_t)\text{Var}(\hat{\mathbf{u}}_t)}}, \tag{8}$$

where $\mathbf{u}_t, \hat{\mathbf{u}}_t \in \mathbb{R}^{N \times c}$ are flattened into vectors prior to computing the correlation. After computing per-timestep Pearson correlations, we quantify long-term stability by measuring the average normalized time up to which the correlation stays above 0.9, denoted as $T^{>0.9}$:

$$T^{>0.9} = \frac{1}{T} \max\left\{t \in [1, T] \mid r(t) \geq 0.9\right\}. \tag{9}$$

This metric reflects the average proportion of the trajectory during which the prediction remains highly corrleated with the ground-truth PDE trajectory.

4.2 Comparisons

We begin by summarizing the performance of our method and the baselines, measured in terms of relative $\mathcal{L}_2$ error and long-term stability $T^{>0.9}$, in Tab. 1. Across all three long-trajectory benchmarks introduced in Sec. 4.1, DiffusionRollout outperforms state-of-the-art neural operators based on regression losses (Li et al., 2021; 2023b; Wu et al., 2024; Hu et al., 2025), including models based on Fourier layers (Li et al.,

	Gray-Scott (Ohana et al., 2024)		Turbulent Flow (Ohana et al., 2024)		Cahn-Hilliard (Soares et al., 2023)		Anisotropic Diffusion (Koehler et al., 2024)	
	Rel $\mathcal{L}_2 \downarrow$	$T^{>0.9} \uparrow$	Rel $\mathcal{L}_2 \downarrow$	$T^{>0.9} \uparrow$	Rel $\mathcal{L}_2 \downarrow$	$T^{>0.9} \uparrow$	Rel $\mathcal{L}_2 \downarrow$	$T^{>0.9} \uparrow$
Regression-Based Neural Solvers								
FNO (Li et al., 2021)	0.652	0.042	0.419	0.526	0.450	0.256	0.035	**0.998**
FactFormer (Li et al., 2023b)	0.620	0.187	0.344	0.744	0.374	0.364	0.250	0.632
Transolver (Wu et al., 2024)	0.430	0.264	0.336	0.773	0.388	0.309	0.405	0.447
Diffusion-Based Neural Solvers								
WDNO (Hu et al., 2025)	0.792	0.022	0.843	0.000	0.567	0.069	1.152	0.000
PDE-Refiner (Lippe et al., 2023) ($s = 1$)	0.409	0.304	0.330	0.799	0.285	0.514	0.048	0.992
PDE-Refiner (Lippe et al., 2023) ($s = 2$)	0.406	0.307	0.321	0.800	0.275	0.543	0.040	0.993
PDE-Refiner (Lippe et al., 2023) ($s = 3$)	0.416	0.303	0.322	0.791	0.267	0.558	0.033	0.994
PDE-Refiner (Lippe et al., 2023) ($s = 4$)	0.431	0.293	0.322	0.833	0.275	0.541	0.031	0.994
PDE-Refiner (Lippe et al., 2023) ($s = 5$)	0.443	0.285	0.322	0.919	0.280	0.531	0.027	0.995
PDE-Refiner (Lippe et al., 2023) ($s = 6$)	0.462	0.275	0.323	0.809	0.289	0.475	0.029	0.995
DiffusionRollout (Ours)	**0.391**	**0.311**	**0.307**	**0.926**	**0.254**	**0.626**	**0.026**	0.996

Table 1: **Quantitative comparison of long rollouts.** For each dataset, we report the relative $\mathcal{L}_2$ error and the average duration of high correlation ($T^{>0.9}$). DiffusionRollout consistently achieves the lowest error and competitive or superior long-term stability, outperforming state-of-the-art neural solvers and a naive autoregressive sampling strategy across various rollout step sizes. **Bold** indicates the best for each column.

	Gray-Scott (Ohana et al., 2024)	Turbulent Flow (Ohana et al., 2024)	Cahn-Hilliard (Soares et al., 2023)
Regression Loss Training	0.479	0.849	0.270
Derivative Threshold	0.395	0.320	0.256
$\hat{\boldsymbol{\epsilon}}_t$ **Threshold**	**0.391**	**0.307**	**0.254**

Table 2: **Quantitative results from the ablation study.** The proposed method achieves the lowest error, outperforming both its regression-trained variant and the version using an alternative surrogate for predictive uncertainty. **Bold** indicates the best for each column.

2021) and transformer architectures tailored for operator learning (Li et al., 2023b; Wu et al., 2024). In addition, DiffusionRollout outperforms recent diffusion-based neural operators (Lippe et al., 2023; Hu et al., 2025), including PDE-Refiner (Lippe et al., 2023), which uses default autoregressive sampling with fixed step sizes, and WDNO (Hu et al., 2025), which leverages wavelet representations to better capture abrupt changes in solutions. We observe a performance drop in the case of WDNO (Hu et al., 2025), which was originally designed to predict entire short-length trajectories at once. This is due to the accumulation of prediction errors, amplified by repeated encoding and decoding through wavelet transforms during long-horizon rollouts.

While PDE-Refiner (Lippe et al., 2023), used as our base model and employing naive autoregressive sampling with fixed step sizes, achieves performance comparable to or better than other baselines, further improvements are possible by adopting our adaptive sampling technique described in Sec. 3.3. In particular, we observe a notable error reduction on the Gray-Scott dataset (Tab. 1, first column), where our adaptive planning results in 15.37% error reduction compared to the base model's fixed-step sampling strategy, which advances 6 steps per iteration. We highlight that this improvement is achieved purely by modifying the inference procedure based on the analysis in Sec. 3.3, with no changes to the training process.

The trend observed in Tab. 1 is further supported by the qualitative results in Fig. 3, which illustrate the effectiveness of the proposed method in capturing the long-term evolution of dynamical systems. The first column of the figure shows the field at an early timestep, while the second column displays the target field at a much later timestep, typically separated by several dozen simulation steps. Despite the apparent differences between the initial and target states, our model generates accurate predictions. In contrast, even state-of-the-art neural solvers such as Transolver (Wu et al., 2024) struggle in this setting, producing noticeably different patterns—especially on the Gray-Scott and Cahn-Hilliard datasets (fifth and sixth columns)—as reflected by the brighter error maps.

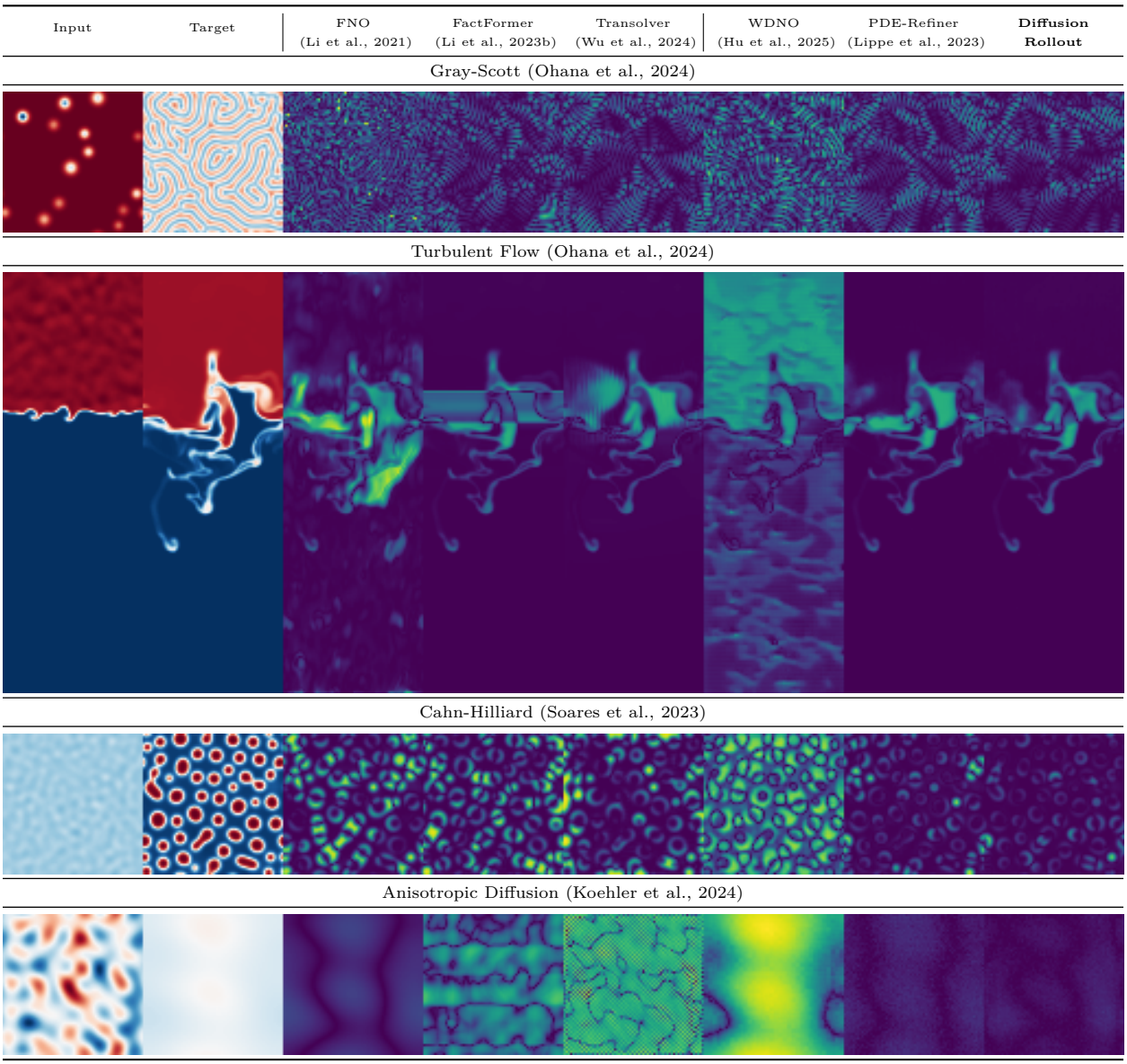

Figure 3: **Qualitative comparison.** Per-pixel error maps (columns 3–8) are visualized for each model's prediction of the target (column 2), given the initial context (column 1). Darker regions in the error maps indicate lower prediction error.

4.3 Ablation Study

We validate our design choices by comparing our method against two variants: (1) the same backbone network trained using the regression loss in Eqn. 2 ("Regression Loss Training"), and (2) an alternative adaptive sampling strategy that uses the temporal derivative computed using the predicted future segment as a proxy for predictive uncertainty ("Derivative Threshold"). Qualitative and quantitative comparisons are summarized in Fig. 4 and Tab. 2, respectively. As shown in the first row of Tab. 2, replacing the diffusion-based training objective with the regression loss in Eqn. 2 results in performance degradation. This is especially pronounced on the Gray-Scott dataset, where the error increases by 8.8%p, highlighting the importance of using a generative training loss. Furthermore, we explored an alternative measure of predictive uncertainty based on the temporal derivative of the predicted solution, motivated by the observation that systems experiencing abrupt changes are more difficult to predict. As shown in Tab. 2, computing uncertainty

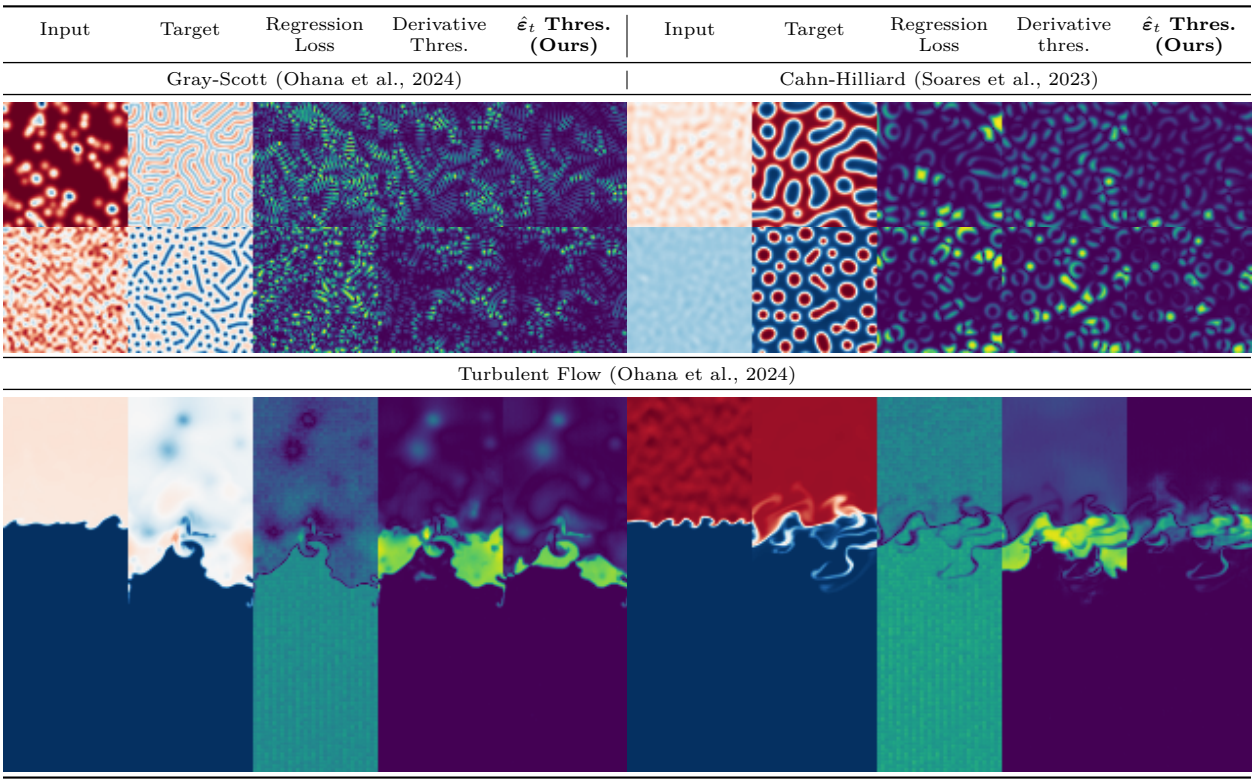

Figure 4: **Qualitative comparison from the ablation study.** Per-pixel error maps (columns 3-5 and 8-10) are shown for each variant of our method, visualizing the prediction error relative to the target (columns 2 and 7), given the initial context (columns 1 and 6). Darker regions in the error maps indicate lower prediction error.

as the standard deviation over multiple samples proves more effective—particularly in the challenging task of predicting turbulent flows.

4.4 Analysis of Hyperparameters

In this section, we present analyses of how different hyperparameter choices affect model performance, evaluated on the test set of the 2D Gray–Scott dataset (Ohana et al., 2024). We begin by examining the time–accuracy trade-off induced by the choice of uncertainty threshold τ. We then demonstrate that the proposed uncertainty estimates, based on standard deviations, perform well even with a small number of samples K. In addition, we analyze the impact of diffusion timesteps S on model accuracy.

Impact of Varying τ. We first explore threshold values $\tau \in \{0.001, 0.01, 0.2, 0.6, 3, 5, 7, 9\}$ other than the default value $\tau = 1$ used for the dataset. As shown in Tab. 3, even with the same model, performance varies with the choice of threshold, highlighting the importance of selecting an appropriate value. Specifically, smaller thresholds make the model more conservative, resulting in smaller steps as reflected by the average step size s_{avg} along the trajectories. In practice, the threshold is selected via binary search based on prediction errors measured on the training set, as described in Sec. 3.3.

Impact of Varying K. Beyond the default number of samples $K = 2$ used throughout the experiments of Sec. 4.2, we additionally evaluate $K \in \{3, 4, 5\}$ to verify how many samples are required for reliable uncertainty estimation. The effect of varying K on prediction accuracy is summarized in Tab. 4. As shown, the relative $\mathcal{L}_2$ error remains consistent across all values of K. In particular, $K = 2$ and $K = 5$ yield nearly

τ	0.001	0.01	0.2	0.6	1	3	5	7	9
Rel $\mathcal{L}_2$	0.399	0.399	0.399	0.396	**0.391**	0.397	0.402	0.406	0.412
s_{avg}	1.000	1.000	1.062	1.087	1.120	1.956	2.300	2.437	2.506

Table 3: **Relative $\mathcal{L}_2$ errors of predicted solutions and average step sizes s_{avg} under different uncertainty thresholds τ.** Choosing an appropriate threshold is crucial for balancing prediction accuracy and inference time, as reflected by both the error and the average step size.

K	2	3	4	5
Rel $\mathcal{L}_2$	**0.391**	0.396	0.395	0.394

Table 4: **Relative $\mathcal{L}_2$ errors of predicted solutions for varying sample counts K.** Reliable uncertainty estimates can be obtained with as few as 2 samples.

S	10	20	30	40	50
Rel $\mathcal{L}_2$	**0.391**	0.399	0.401	0.401	0.402

Table 5: **Relative $\mathcal{L}_2$ errors of predicted solutions for different diffusion timesteps.** Increasing the diffusion timesteps during generation, and thus the NFEs, results in higher errors.

identical results, suggesting that only a small number of samples is sufficient to obtain reliable uncertainty estimates.

Impact of Varying Diffusion Timesteps. In our experiments, we employ the DDIM sampler (Song et al., 2022) with $S = 10$ timesteps to generate trajectories. To further examine the effect of the number of sampling steps on model performance, we also evaluate the model with larger values $S \in \{20, 30, 40, 50\}$. As shown in Tab. 5, increasing the number of diffusion timesteps—and thus the number of function evaluations (NFEs)—does not necessarily improve accuracy; instead, it slightly increases prediction error, aligning with our discussion in Sec. 3.3.

5 Conclusion

We proposed DiffusionRollout, a diffusion-based neural PDE solver designed to predict long rollouts of physical systems governed by PDEs. Building on the benefits of generative modeling for PDE modeling, we introduce a method that leverages the norm of standard deviations—computed from multiple samples drawn from a pre-trained diffusion model—as a surrogate measure of predictive uncertainty. This uncertainty estimate is then used during the autoregressive sampling process to reject erroneous predictions that could otherwise propagate and significantly degrade rollout accuracy. In experiments on three challenging benchmark problems, the proposed method outperforms state-of-the-art baselines and demonstrates clear performance gains over conventional autoregressive sampling, which, by contrast, does not account for potential prediction errors during inference.

Limitations and Societal Impacts. While DiffusionRollout improves long-term prediction accuracy and inherits key benefits of generative modeling, it relies on an iterative sampling process, which is slower than regression-based neural operators. A promising direction for future work is to integrate recent one-step or few-step diffusion models into our adative rollout framework for improved inference efficiency. Another interesting direction is to explore learning a dedicated planner module that adjusts the uncertainty threshold based on the current context. Our work takes a step toward applying generative modeling to a broad range of domains in scientific computing, with the potential to accelerate scientific discovery.

References

Benedikt Alkin, Andreas Fürst, Simon Schmid, Lukas Gruber, Markus Holzleitner, and Johannes Brandstetter. Universal physics transformers. In *NeurIPS*, 2024.

Kushal Arora, Layla El Asri, Hareesh Bahuleyan, and Jackie Cheung. Why exposure bias matters: An imitation learning perspective of error accumulation in language generation. In *Findings of the Association for Computational Linguistics*, 2022.

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. An actor-critic algorithm for sequence prediction. In *ICLR*, 2017.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *NeurIPS*, 2015.

Lucas Berry, Axel Brando, and David Meger. Shedding light on large generative networks: Estimating epistemic uncertainty in diffusion models. In *Uncertainty in Artificial Intelligence*, 2024.

Shuhao Cao. Choose a transformer: fourier or galerkin. In *NeurIPS*, 2021.

Matthew A Chan, Maria J Molina, and Christopher A Metzler. Estimating epistemic and aleatoric uncertainty with a single model. In *NeurIPS*, 2024.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. In *NeurIPS*, 2024.

Yu Chen, Lingfei Wu, and Mohammed J. Zaki. Reinforcement learning based graph-to-sequence model for natural question generation. In *ICLR*, 2020.

Woojin Cho, Minju Jo, Haksoo Lim, Kookjin Lee, Dongeun Lee, Sanghyun Hong, and Noseong Park. Parameterized physics-informed neural networks for parameterized pdes. In *ICML*, 2024.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking attention with performers. In *ICLR*, 2021.

Stefan Depeweg, José Miguel Hernández-Lobato, Finale Doshi-Velez, and Steffen Udluft. Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning. In *ICML*, 2018.

Drummond B Fielding, Eve C Ostriker, Greg L Bryan, and Adam S Jermyn. Multiphase gas and the fractal nature of radiative turbulent mixing layers. *The Astrophysical Journal Letters*, 894(2):L24, 2020.

J. Gawlikowski, C. R. N. Tassi, M. Ali, et al. A survey of uncertainty in deep neural networks. *Artificial Intelligence Review*, 56(Suppl 1):1513–1589, 2023. doi: 10.1007/s10462-023-10562-9. URL `https://doi.org/10.1007/s10462-023-10562-9`.

Ernst Hairer, Gerhard Wanner, and Syvert P. Nørsett. *Solving Ordinary Differential Equations I*. Springer, 1987.

Jian Han, Jinlai Liu, Yi Jiang, Bin Yan, Yuqi Zhang, Zehuan Yuan, Bingyue Peng, and Xiaobing Liu. Infinity: Scaling bitwise autoregressive modeling for high-resolution image synthesis. In *CVPR*, 2024.

Zhongkai Hao, Chengyang Ying, Zhengyi Wang, Hang Su, Yinpeng Dong, Songming Liu, Ze Cheng, Jun Zhu, and Jian Song. Gnot: A general neural operator transformer for operator learning. In *ICML*, 2023.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

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 *ICLR*, 2025.

Jiahe Huang, Guandao Yang, Zichen Wang, and Jeong Joon Park. Diffusionpde: Generative pde-solving under partial observation. In *NeurIPS*, 2024.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *ICLR*, 2020.

Felix Koehler, Simon Niedermayr, Nils Thuerey, et al. Apebench: A benchmark for autoregressive neural emulators of pdes. *Advances in Neural Information Processing Systems*, 37:120252–120310, 2024.

Aditi S. Krishnapriyan, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael W Mahoney. Characterizing possible failure modes in physics-informed neural networks. In *NeurIPS*, 2021.

Thorsten Kurth, Shashank Subramanian, Peter Harrington, Jaideep Pathak, Morteza Mardani, David Hall, Andrea Miele, Karthik Kashinath, and Anima Anandkumar. Fourcastnet: Accelerating global high-resolution weather forecasting using adaptive fourier neural operators. In *Proceedings of the Platform for Advanced Scientific Computing Conference*, 2023.

Wilhelm Kutta. *Beitrag zur näherungsweisen Integration totaler Differentialgleichungen.* Teubner, 1901.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NeurIPS*, 2017.

Rémi Leblond, Jean-Baptiste Alayrac, Anton Osokin, and Simon Lacoste-Julien. SEARNN: Training RNNs with global-local losses. In *ICLR*, 2018.

Zijie Li, Kazem Meidani, and Amir B. Farimani. Transformer for partial differential equations' operator learning. 2023a.

Zijie Li, Dule Shu, and Amir Barati Farimani. Scalable transformer for pde surrogate modeling. In *NeurIPS*, 2023b.

Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural Operator: Graph Kernel Network for Partial Differential Equations. *arXiv*, 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 *ICLR*, 2021.

Zongyi Li, Miguel Liu-Schiaffini, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Learning chaotic dynamics in dissipative systems. In *NeurIPS*, 2022.

Zongyi Li, Nikola Borislavov Kovachki, Chris Choy, Boyi Li, Jean Kossaifi, Shourya Prakash Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, and Anima Anandkumar. Geometry-Informed Neural Operator for Large-Scale 3D PDEs. In *NeurIPS*, 2023c.

Zongyi Li, Daniel Zhengyu Huang, Burigede Liu, and Anima Anandkumar. Fourier neural operator with learned deformations for pdes on general geometries. *JMLR*, 2024.

Phillip Lippe, Bastiaan S. Veeling, Paris Perdikaris, Richard E Turner, and Johannes Brandstetter. PDE-Refiner: Achieving Accurate Long Rollouts with Temporal Neural PDE Solvers. In *NeurIPS*, 2023.

Ruben Ohana, Michael McCabe, Lucas Thibaut Meyer, Rudy Morel, Fruzsina Julia Agocs, Miguel Beneitez, Marsha Berger, Blakesley Burkhart, Stuart B. Dalziel, Drummond Buschman Fielding, Daniel Fortunato, Jared A. Goldberg, Keiya Hirashima, Yan-Fei Jiang, Rich Kerswell, Suryanarayana Maddu, Jonah M. Miller, Payel Mukhopadhyay, Stefan S. Nixon, Jeff Shen, Romain Watteaux, Bruno Régaldo-Saint Blancard, François Rozet, Liam Holden Parker, Miles Cranmer, and Shirley Ho. The well: a large-scale collection of diverse physics simulations for machine learning. In *NeurIPS*, 2024.

OpenAI. Gpt-4 technical report. *arXiv*, 2024.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, 2023.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. In *ICLR*, 2016.

David Ruhe, Jonathan Heek, Tim Salimans, and Emiel Hoogeboom. Rolling diffusion models. In *ICML*, 2024.

Carl Runge. Über die numerische auflösung von differentialgleichungen. *Mathematische Annalen*, 1895.

Aliaksandra Shysheya, Cristiana Diaconu, Federico Bergamin, Paris Perdikaris, José Miguel Hernández-Lobato, Richard E. Turner, and Emile Mathieu. On conditional diffusion models for pde simulations. In *NeurIPS*, 2024.

Elvis do A. Soares, Amaro G. Barreto, and Frederico W Tavares. Exponential Integrators for Phase-Field Equations using Pseudo-spectral Methods: A Python Implementation. *arXiv*, 2023.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2022.

Kiwhan Song, Boyuan Chen, Max Simchowitz, Yilun Du, Russ Tedrake, and Vincent Sitzmann. History-guided video diffusion. *arXiv*, 2025.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *NeurIPS*, 2019.

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

Keyu Tian, Yi Jiang, Zehuan Yuan, BINGYUE PENG, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. In *NeurIPS*, 2024.

Sifan Wang and Paris Perdikaris. Long-time integration of parametric evolution equations with physics-informed deeponets. *J. Comput. Phys.*, 2023.

Tian Wang and Chuang Wang. Latent neural operator for solving forward and inverse pde problems. In *NeurIPS*, 2024.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. Neural text generation with unlikelihood training. In *ICLR*, 2020.

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for pdes on general geometries. In *ICML*, 2024.

Zipeng Xiao, Zhongkai Hao, Bokai Lin, Zhijie Deng, and Hang Su. Improved operator learning by orthogonal attention. In *NeurIPS*, 2024.

Yifan Xu, Kening Zhang, Haoyu Dong, Yuezhou Sun, Wenlong Zhao, and Zhuowen Tu. Rethinking exposure bias in language modeling. *arXiv preprint arXiv:1910.11235*, 2019.

Wen Zhang, Yang Feng, Fandong Meng, Di You, and Qun Liu. Bridging the gap between training and inference for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

Appendix

In this technical appendix, we provide implementation details (Sec. A.1), along with quantitative comparisons of adaptive planning applied to regression-based models (Sec. A.2) and additional qualitative results (Sec. A.3).

A.1 Implementation Details

We implement our diffusion-based neural solver, including PDE-Refiner Lippe et al. (2023), based on the DiT Peebles & Xie (2023) architecture with 3D attention layers with rotary embeddings to jointly capture temporal and spatial dependencies. Instead of designing a separate module for processing the conditoning input $\mathbf{u}_C$, we simply concatenate the full sequence $[\mathbf{u}_C \| \mathbf{u}_F]$ and feed it into the 3D self-attention layers, along with an indicator that speifies which frames are conditions. To enable this, we adopt the training objective from Diffusion Forcing Chen et al. (2024); Song et al. (2025), which assigns an independent noise level to each frame. This allows us to easily designate condition frames by assigning them a noise level of 0, while applying nonzero noise levels to the rest. At inference time, we employ DDIM Song et al. (2022) sampling with 10 steps.

To compute the standard deviations for uncertainty quantification, we draw two samples ($K = 2$), which can be efficiently obtained by batching the inputs—incurring no significant computational overhead. We use the uncertainty threshold of 1, 1, 5, and 0.32 for Gray-Scott Reaction-Diffusion Equation, Cahn-Hilliard Equation, 2D Turbulent Radiative Flow Equation, and Anisotropic Diffusion respectively.

For all regression-based neural operators, including FNO Li et al. (2021), FactFormer Li et al. (2023b), and Transolver Wu et al. (2024), we use the implementations provided by Wu et al. (2024), which faithfully replicate the original models.

A.2 Results of Varying Step Sizes on Regression-Based Neural Solvers

We also investigate the impact of varying step sizes in regression-based neural operators. As shown in the trends in Fig. A5, the relative error generally increases as the step size decreases. We attribute this to regression-based models being more susceptible to network approximation error, where a larger number of NFEs—caused by smaller step sizes—leads to greater error accumulation during long-horizon rollouts. Note that the trajectories predicted by FNO Li et al. (2021) diverged and yielded invalid numerical values when smaller step sizes were used, as indicated by $\times$ markers in the plots. Furthermore, as shown in Sec. 4.2, regression-based methods already underperform compared to diffusion-based approaches, and our adaptive rollout strategy is only applicable to the latter.

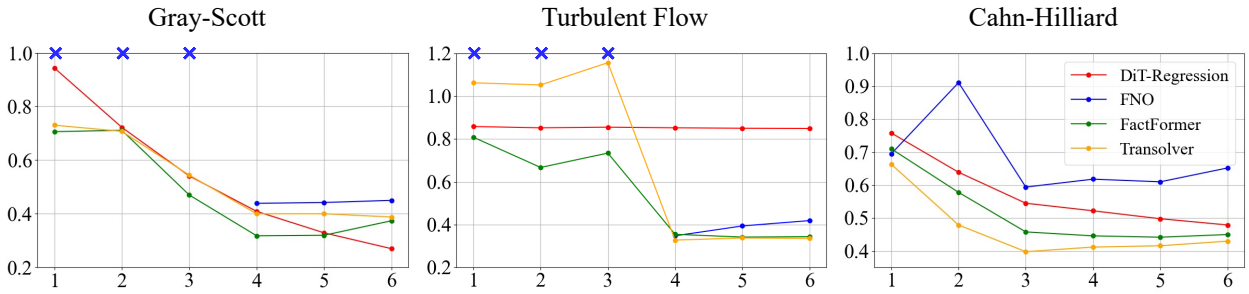

Figure A5: **Relative $\mathcal{L}_2$ errors of regression-based models with varying autoregressive step sizes.**

The section for additional qualitative results are in the following pages.

A.3 Additional Qualitative Results

We present additional qualitative results in Fig. A6, where DiffusionRollout produces the most accurate predictions, as visually indicated by the darker error maps.

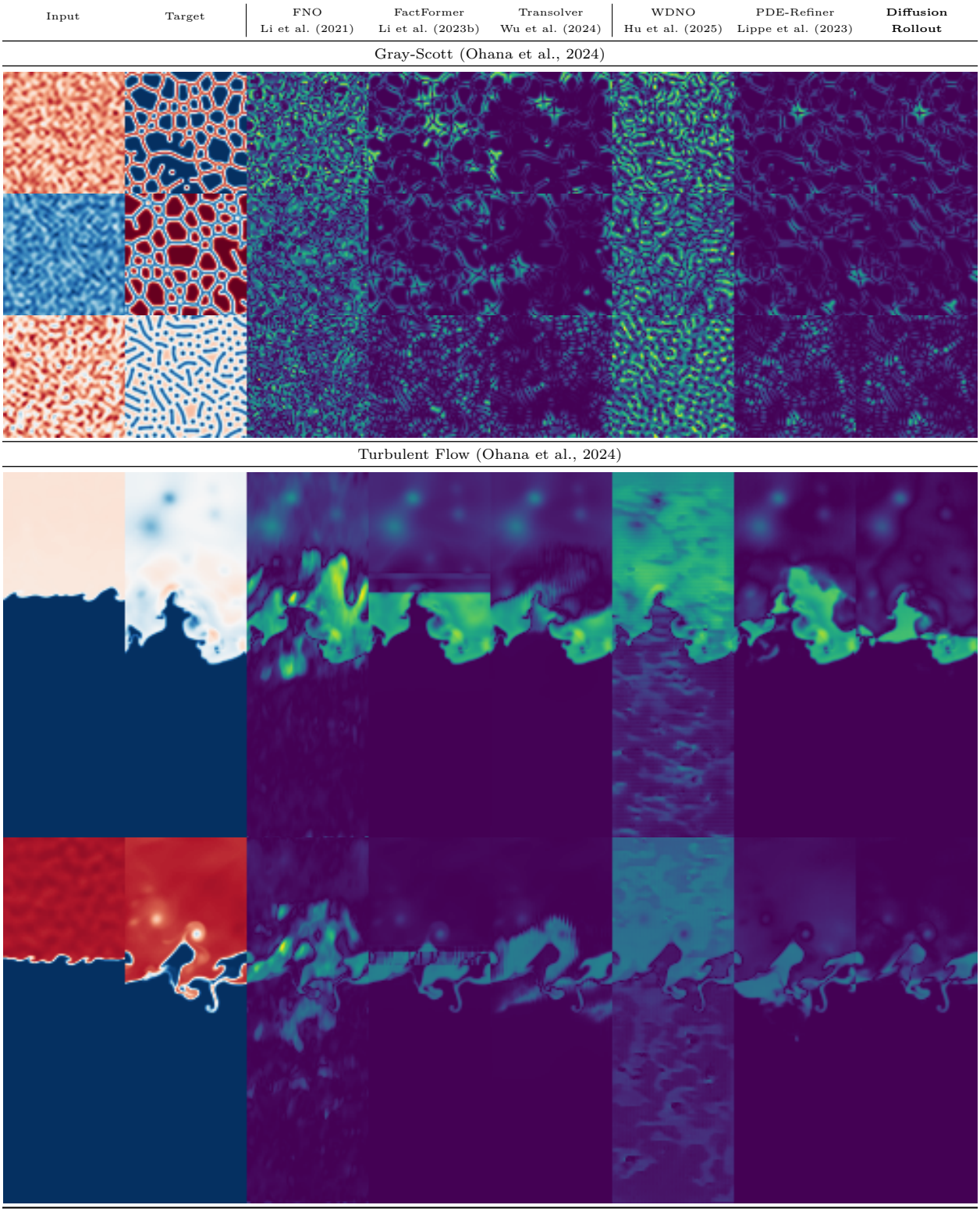

Figure A6: **Additional Qualitative Results.**

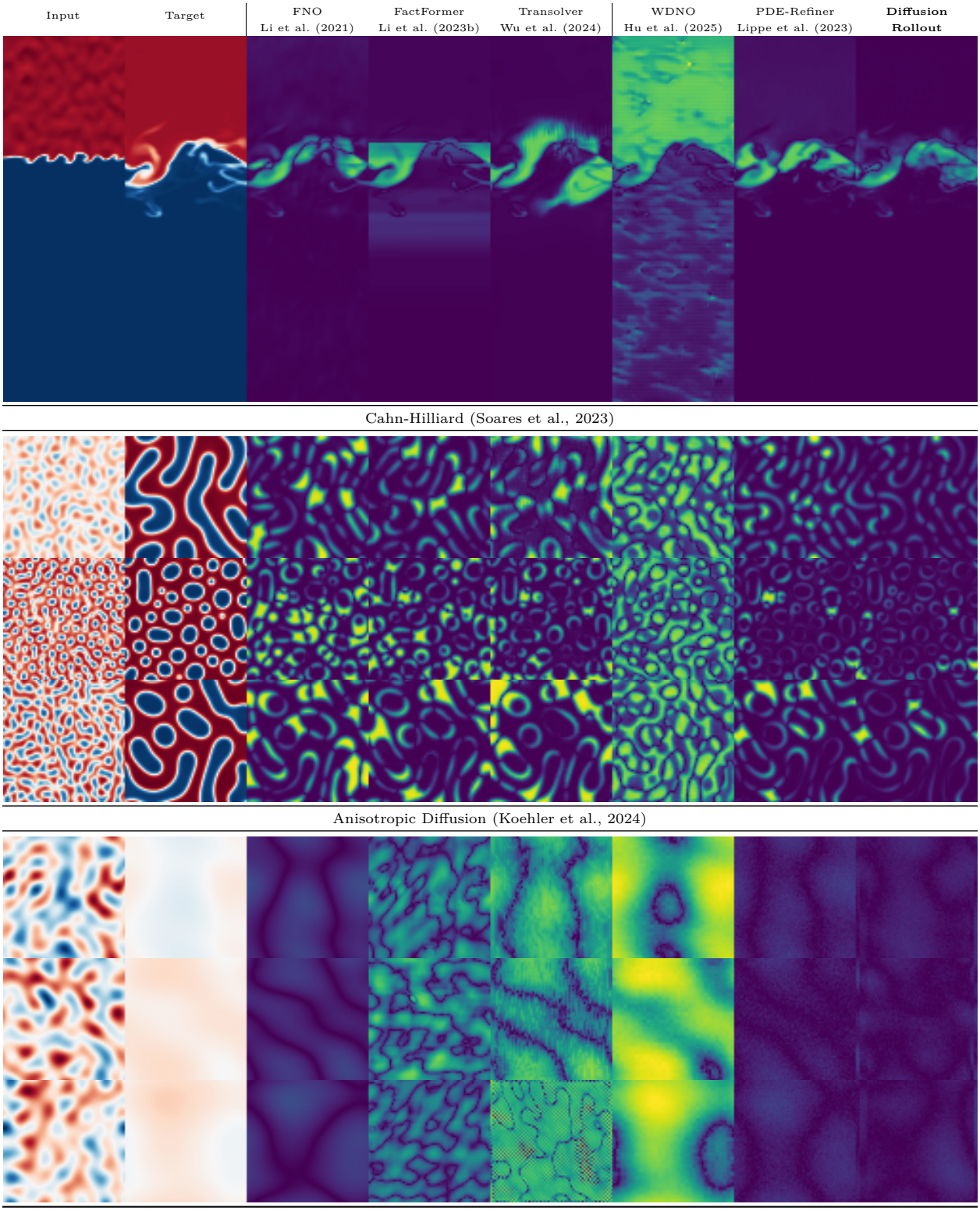

Figure A6: **Additional Qualitative Results.**