AUTOPDEBENCH: BENCHMARKING LLM AUTO-RESEARCH FOR NEURAL PDE SOLVER DESIGN

Anonymous authors
Paper under double-blind review

ABSTRACT

Partial differential equations (PDEs) are essential for modeling complex physical systems, and neural solvers have recently emerged as powerful data-driven tools for numerically solving them. However, existing neural solvers struggle with domain-specific challenges, such as varying parameters and high-speed flows, necessitating specialized architectures. Manually designing these specialized solver architectures is a highly iterative, time-consuming process requiring deep expertise, creating a significant bottleneck in scientific discovery. We propose leveraging autonomous AI research agents to automate the synthesis of specialized solvers. To support this, we introduce AutoPDEBench, a benchmark dedicated to LLM-driven automated research for PDE solver design. The benchmark includes 25 challenging datasets featuring both novel and actively studied physical scenarios. We evaluate a suite of general-purpose models (transformer, ROM, and graph-based) alongside a multi-agent instantiation of the iterative automated research pipeline, which serves as an agentic baseline. Empirical results show that the iterative automated research system significantly outperforms the general-purpose neural solver baselines. Our findings demonstrate the viability of using AI agents to automatically design neural solvers for complex physical systems. AutoPDEBench provides a foundational testbed to accelerate agent-driven scientific discovery in physics and engineering. Code and datasets are fully open-sourced at `https://anonymous.4open.science/r/AutoPDEBench-DF0B/README.md`.

1 INTRODUCTION

Partial differential equations (PDEs) are the foundational language of physics. They provide the essential mathematical framework to describe how complex systems evolve over space and time. Recent advancements have seen neural solvers demonstrate remarkable performance in learning PDE solutions purely from data. While these models achieve strong results across many standard datasets, others present domain-specific challenges that necessitate specialized architectures. For example, when evaluating datasets with mixed PDE parameters, standard neural solvers cannot reliably map initial conditions to target trajectories; the lack of explicit parameter inputs renders the predictions ill-posed. Encountering this exact scenario, Takamoto et al. (2023) recognized that solvers must explicitly encode the underlying PDE parameters. To address this, they introduced a parameter-guided channel attention mechanism that conditions the neural solver on these varying physical coefficients. High-speed flows present another domain-specific challenge. Because these systems involve extreme gradients and discontinuities, such as shock waves, existing neural solvers typically suffer from numerical instability or non-physical smoothing. To overcome this, Helwig et al. (2026) introduced NeuralCFL, a framework that explicitly predicts an adaptive timestep size to stabilize the simulation during rapid state changes.

Traditionally, human researchers invest a substantial amount of time designing these domain-specific solvers, a process that demands deep specialized expertise and significant computational resources. Because this manual engineering is highly iterative and uniquely tailored to each physical system, it severely limits the pace and scalability of scientific discovery. This critical bottleneck motivates the development of autonomous AI research agents capable of automatically synthesizing specialized solver pipelines. By dynamically adapting to complex physical dynamics, these agents promise to accelerate the modeling process and reduce the reliance on human intervention.

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

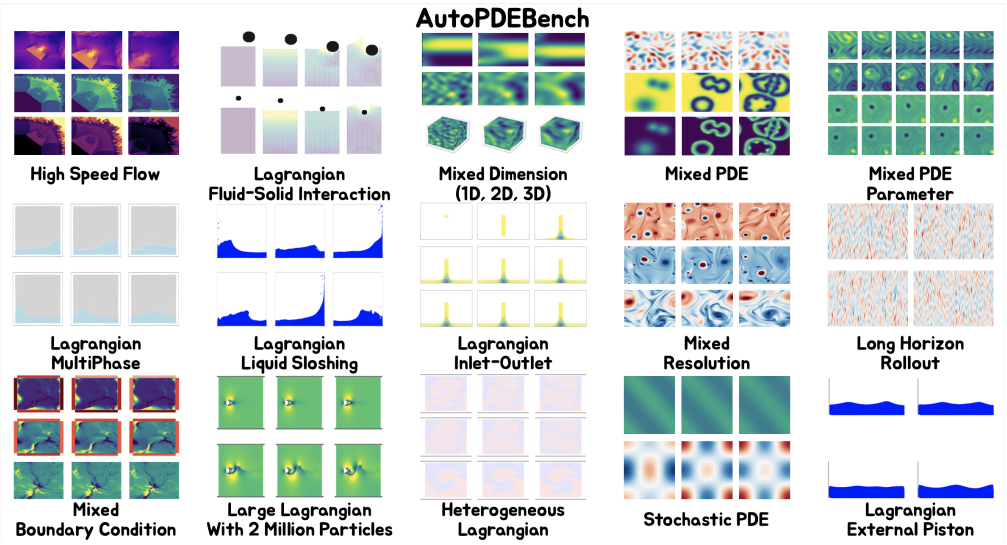


Figure 1: Overview of the example datasets proposed by AutoPDEBench. AutoPDEBench consists of 25 challenging datasets, each presenting specific domain challenges and necessitating specialized solvers. It features both entirely novel and currently understudied datasets. We intentionally reduced the resolution of all figures to maintain a reasonable file size. High-quality versions are available in our GitHub repository.

To this end, we introduce AutoPDEBench, a benchmark dedicated to LLM-driven automated research for designing specialized solvers to tackle domain-specific challenges. AutoPDEBench comprises 25 challenging datasets, encompassing both actively studied and entirely novel physical scenarios. These datasets present distinct, idiosyncratic challenges that general-purpose neural solvers either struggle with or fail to handle entirely, thereby necessitating specialized architectures. To establish robust baselines, we first evaluate a suite of versatile general-purpose models, including transformer-based, reduced-order modeling (ROM), and graph-based methods. We then introduce a multi-agent instantiation of the iterative automated research pipeline to serve as an agentic baseline. Our empirical results demonstrate that this auto-research system achieves superior performance compared to the general solver baselines, underscoring the immense potential of utilizing AI agents to automatically design neural solvers for complex physical systems. However, outperforming general-purpose baselines does not mean the task is fully solved. In classical numerical analysis, the search for schemes with minimal local truncation error is an ongoing pursuit, as per-step errors inevitably accumulate over long time horizons and are often amplified by nonlinear dynamics. Neural solvers face this exact vulnerability, with single-step prediction errors compounding severely during autoregressive rollouts. We therefore position AutoPDEBench as an open-ended testbed designed to continually challenge agents to discover increasingly accurate solvers.

Our contributions are as follows:

- **Problem Identification:** We highlight that the domain of neural PDE solvers encompasses a diverse array of physical systems characterized by idiosyncratic challenges. Traditionally, addressing these unique constraints has demanded extensive manual effort to design specialized architectures.
- **Proposed Benchmark:** We introduce AutoPDEBench, the first comprehensive benchmark featuring 25 challenging datasets with unique physical constraints. This collection spans both actively researched problems and entirely novel domains introduced to the community for the first time.
- **Empirical Analysis:** We conduct extensive evaluations comparing versatile, general-purpose neural PDE solvers against a multi-agent instantiation of the iterative automated research pipeline. Our results demonstrate the profound potential of utilizing LLM agents to autonomously design specialized solver pipelines for complex physical systems.

2 Related Work

We identify three key areas closely related to our work.

Neural PDE Solvers. The evolution of neural PDE solvers originated with early architectures designed to learn solution operators on regular Euclidean grids (Lu et al., 2021; Li et al., 2021; 2020; Kovachki et al., 2023; Rahman et al., 2023; Tripura & Chakraborty, 2023; Tran et al., 2023). To overcome the geometric limitations of uniform grid-based representations, subsequent research rapidly expanded model capabilities to accommodate unstructured meshes (Pfaff et al., 2021; Boussif et al., 2022; Li et al., 2023; Han et al., 2022; Lino et al., 2021) and Lagrangian particle-based simulations (Sanchez-Gonzalez et al., 2020; Toshev et al., 2024b; 2023; Prantl et al., 2022; Rochman-Sharabi et al., 2025). As these methodologies matured, the community introduced comprehensive benchmark suites (Wang et al., 2026a; Toshev et al., 2024a; Gupta & Brandstetter, 2022; Ohana et al., 2024; Koehler et al., 2024; Takamoto et al., 2024) to establish standardized evaluation strategies and curate large-scale datasets across a wide variety of physical systems. Building upon these established baselines, recent solvers have sought to push performance boundaries across multiple complex axes. These advancements include architectures capable of generalizing across varying PDE parameters (Takamoto et al., 2023; Brandstetter et al., 2022) and diverse boundary conditions (Horie & MITSUME, 2022; Wang et al., 2024; Saad et al., 2023), handling highly complex geometries (Wu et al., 2024; Luo et al.; Wen et al., 2025; Mousavi et al., 2025), adapting seamlessly across 1D, 2D, and 3D spatial dimensions (Chen et al., 2025), and unifying Eulerian field and Lagrangian particle representations (Alkin et al., 2024; 2025). Most recently, this trajectory has culminated in the exploration of physics foundation models (Hao et al., 2024; McCabe et al., 2024; Ye et al., 2024; Herde et al., 2024; Shen et al., 2024; Zhou et al.; Wang et al., 2025), which are pretrained on massive, heterogeneous datasets to achieve strong zero-shot and few-shot generalization capabilities across unseen dynamic regimes.

Beyond forward temporal simulation, neural network methodologies have been widely adopted across adjacent domains in scientific machine learning. These broader applications include solving inverse problems (Li et al., 2025b; Zheng et al., 2025; Molinaro et al., 2023; Li et al., 2025a; Wang & Wang, 2024) and developing hybrid frameworks that integrate neural networks directly into classical numerical schemes to enhance their performance and mitigate truncation errors (List et al., 2022; Sun et al., 2023; Greenfeld et al., 2019; Sappl et al., 2019).

LLM for PDE Solvers. Researchers have extensively utilized LLMs to autonomously generate traditional numerical PDE solvers (Li et al., 2025c; Wu et al., 2025; Hang et al., 2026; Dong et al., 2026). These approaches typically translate high-level mathematical formulations and boundary conditions directly into executable simulation scripts. This automation significantly reduces the manual programming effort and tedious debugging traditionally required for complex scientific computing. Recent studies have extended this paradigm to neural architectures. For example, Wuwu et al. (2025); Song et al. (2026) leverage LLM agents to design neural solvers; however, their evaluations remain confined to standard benchmark datasets. Our work diverges significantly by deploying LLM agents to tackle complex datasets characterized by unique physical constraints that require domain-specific solvers. Because our benchmark introduces entirely novel, previously unstudied physical systems, the LLM cannot simply retrieve memorized templates from its pre-training data. Instead, the agent must autonomously identify the underlying physical bottlenecks and synthesize novel architectural solutions to overcome them.

LLM Auto Research. The integration of Large Language Models (LLMs) into the scientific workflow has evolved along a spectrum of increasing autonomy. Initially, research focused on utilizing LLMs as assistive tools to augment human effort in specific, localized tasks, such as conducting literature reviews (Agarwal et al., 2025; Gao et al., 2025; Skarlinski et al., 2024; Go et al., 2025), generating preliminary ideas (Yang et al., 2025; Qiu et al., 2025; Ghafarollahi & Buehler, 2024), and assisting with experimental planning and iteration (Ferreira et al., 2026; Guo et al., 2026). As their capabilities expanded, LLMs transitioned into more autonomous roles, functioning as analytical agents capable of complex information processing with minimal human intervention. In this capacity, they have been deployed for automated machine learning task modeling (Huang et al., 2023; Li et al., 2024a; Chan et al., 2025) and the identification of governing equations from observational data (Chen et al., 2026; Merler et al., 2024). Most recently, the field has advanced toward the LLM scientists paradigm. These systems operate as general-purpose, active agents capable of or-

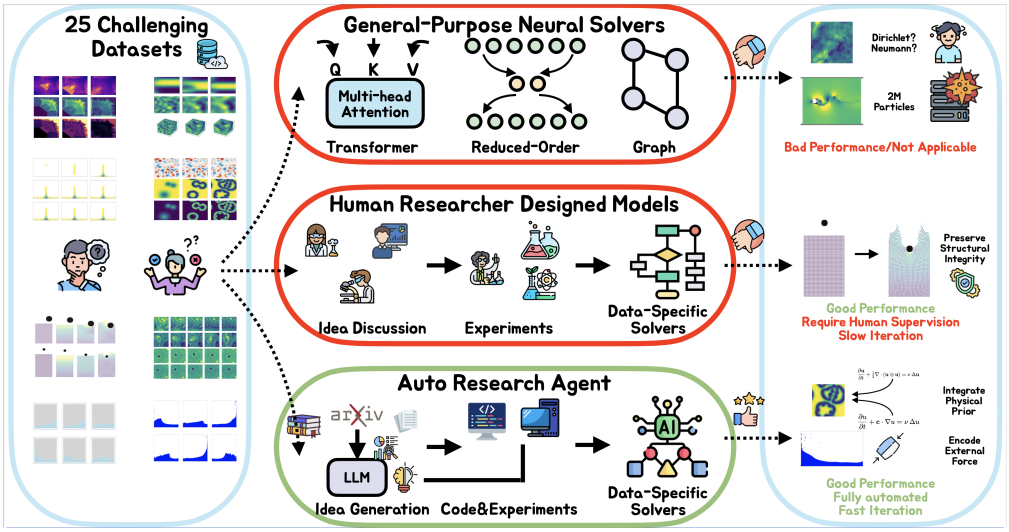


Figure 2: Overall framework of AutoPDEBench. We propose 25 challenging datasets, each presenting unique domain-specific challenges that necessitate specialized solvers. While general-purpose neural solvers can often be applied, they inherently lack the mechanisms to handle these specific constraints, leading to suboptimal performance. Traditionally, researchers manually design specialized solvers for these tasks, which requires extensive domain expertise and is highly time-consuming. We argue that automated research agents offer a compelling alternative, leveraging LLMs to autonomously design neural solvers capable of addressing these dataset-specific challenges.

chestrating the entire scientific discovery process end-to-end (Mitchener et al., 2025; Novikov et al., 2025; Brixi et al., 2026; Ghareeb et al., 2025; Tang et al., 2025; Schmidgall et al., 2025). By independently navigating from novel ideation and autonomous experimental execution to synthesizing genuine findings into full research paper drafts, these pipelines represent a significant step toward fully autonomous natural science discovery.

3 AutoPDEBench

3.1 Problem Formulation

We present the overall framework in Figure 2. We consider the simulation of a physical system where our goal is to learn a neural simulator Φ_θ that predicts a future output state $y \in \mathcal{Y}$ from an input state $x \in \mathcal{X}$. The model also accepts optional supplementary information $c \in \mathcal{C}$, which can include a variable timestep Δt, boundary conditions, PDE parameters, or the governing PDE equations. Thus, the simulator is defined as $\Phi_\theta : \mathcal{X} \times \mathcal{C} \to \mathcal{Y}$, yielding the prediction rule:

$$y = \Phi_\theta(x, c). \tag{1}$$

Each dataset is split into disjoint $\mathcal{D}_{\text{train}}$, $\mathcal{D}_{\text{val}}$, and $\mathcal{D}_{\text{test}}$, and $\mathcal{L}(\Phi_\theta; \mathcal{D})$ denotes evaluation metrics evaluated on split $\mathcal{D}$. We evaluate models under a strict hardware constraint. Given a fixed GPU with memory budget B, our objective is to achieve the best possible performance. We place no restrictions on training or inference time, FLOPs, or parameter count.

General-purpose solver. Suppose $h^\star$ is the optimal hyperparameter configuration evaluated on the validation dataset that respects the memory budget B:

$$h^\star = \underset{h\,:\,\text{Mem}(\Phi,h)\le B}{\arg\min} \mathcal{L}\big(\Phi_{\theta_h}; \mathcal{D}_{\text{val}}\big), \qquad \theta_{h^\star} = \text{Train}(\Phi, h^\star, \mathcal{D}_{\text{train}}). \tag{2}$$

We then use this configuration to report the final test performance $\mathcal{L}(\Phi_{\theta_{h^\star}}; \mathcal{D}_{\text{test}})$.

Auto-research agents. We abstract the research process as an agentic system $\mathcal{A}$ that, at each round, maps a context $\mathcal{I}$ to a candidate solution. The context $\mathcal{I}$ aggregates the information available to the agent, such as the dataset description and the modeling objective. The agentic system is a mapping

Dataset	Description	Source
`HighSpeedFlow` §I.1	Supersonic flows that exceed the speed of sound and create sudden changes such as shock waves.	Public
`RealData` §I.2	Real-world data consists of noisy measurements collected using time-resolved PIV.	Public
`MixedParam` §I.3	Datasets with mixed PDE parameters, where the governing coefficients vary across trajectories.	Ours
`Heterogeneous` §I.4	Lagrangian particle datasets with particles possessing distinct physical attributes such as density.	Ours
`Partial` §I.5	Datasets with partially observed initial conditions that require predicting the full trajectory.	Ours
`MixedDomain` §I.6	Datasets featuring grid, mesh, and particle representations of the same underlying physical system.	Ours
`MultiPhysics` §I.7	Multiphysics simulations that capture the interactions between multiple physical processes.	Public
`3D` §I.8	Spatiotemporal datasets featuring three-dimensional volumetric observations.	Public
`IrregularTime` §I.9	Datasets featuring observations recorded at non-uniform time intervals.	Ours
`LagInletOutlet` §I.10	Lagrangian inlet-outlet data lacking 1-to-1 particle correspondence.	Ours
`LagFSI` §I.11	Lagrangian fluid-solid interaction datasets where both phases are represented by Lagrangian particles.	Ours
`Extreme` §I.12	Datasets with extreme scale capturing 3D supernova blastwaves across seven orders of magnitude.	Public
`MixedRes` §I.13	Generalization datasets that capture PDE dynamics on diverse domains and resolutions.	Ours
`ExtLag1` §I.14	Lagrangian particle datasets with liquid sloshing as external force.	Ours
`ExtLag2` §I.15	Datasets capturing Lagrangian fluid dynamics driven by the external force of a moving piston.	Ours
`MixedDim` §I.16	Datasets capturing identical PDE dynamics in 1D, 2D, and 3D.	Ours
`MixedBC` §I.17	Datasets capturing identical PDE under Dirichlet, Neumann, and periodic boundary conditions.	Ours
`LongHorizon` §I.18	Datasets featuring extremely long-horizon, 5,000-step rollouts.	Ours
`Stochastic` §I.19	Stochastic PDE dataset	Ours
`MultiPhase` §I.20	Lagrangian particle datasets modeling multiphase liquid-gas interactions.	Ours
`MixedPDE` §I.21	Datasets comprising a diverse range of PDEs,	Ours
`LargeLag` §I.22	Large-scale, high-resolution Lagrangian dataset featuring millions of particles.	Ours
`Sim2Real` §I.23	Datasets designed for sim-to-real transfer.	Public
`HighDimensional` §I.24	5D gyrokinetic dataset capturing high-dimensional PDE dynamics.	Ours
`ComplexGeometry` §I.25	Datasets featuring flow around complex geometries.	Public

Table 1: Overview of datasets in AutoPDEBench. Detailed descriptions and visualizations for each dataset are provided in their individual Appendix section.

$\mathcal{A} : \mathcal{I} \to \mathcal{M} \times \mathcal{F} \times \mathcal{H}$ producing a model file $m \in \mathcal{M}$ (architecture Φ_m), a loss file $\ell \in \mathcal{F}$ (training objective $\mathcal{L}^{\text{train}}$, which governs optimization but does not alter the fixed evaluation metrics), and a config file $h \in \mathcal{H}$ (hyperparameters). Over rounds $i = 1, \ldots, N$, the context $\mathcal{I}_i$ is updated with the latest history, and the agent produces

$$(m_i, \ell_i, h_i) = \mathcal{A}(\mathcal{I}_i), \qquad \theta_i = \text{Train}\big(\Phi_{m_i}, \mathcal{L}^{\text{train}}, h_i, \mathcal{D}_{\text{train}}\big), \tag{3}$$

subject to the memory limit $\text{Mem}(\Phi_{m_i}, h_i) \leq B$. Each candidate is trained on $\mathcal{D}_{\text{train}}$ under its generated objective $\mathcal{L}^{\text{train}}$ and hyperparameters h_i, and the agent observes only the validation outcome $r_i = \mathcal{L}(\Phi_{m_i,\theta_i}; \mathcal{D}_{\text{val}})$, never any quantity on $\mathcal{D}_{\text{test}}$. After N rounds, it commits its best validated configuration $i^\star = \arg\min_i r_i$, and the reported performance is its evaluation on the held-out test split, $\mathcal{L}(\Phi_{m_{i^\star},\theta_{i^\star}}; \mathcal{D}_{\text{test}})$.

3.2 Datasets

We propose 25 challenging datasets, each presenting unique constraints that necessitate domain-specific solvers. This collection includes both entirely novel and currently understudied physical systems. Example visualizations and overarching descriptions are provided in Figure 1 and Table 1, respectively. For comprehensive details regarding data generation, governing physics, problem setups, and additional visualizations, we refer readers to the dedicated appendices for each dataset.

3.3 Evaluation Protocol

Our evaluation protocol assesses final model performance on the test dataset. Given our GPU resource constraints, each model was tuned to maximize performance within these limits. Our primary evaluation metric is RMSE. We also use dataset-specific metrics to evaluate additional aspects such as physical consistency and practical utility. The additional evaluation results for each dataset are provided in their individual appendix sections, while detailed descriptions of these supplementary metrics can be found in Appendix C.

3.4 General-Purpose Neural Solver Baselines

To establish our baselines, we consider transformer-based, reduced-order modeling, and graph-based neural solvers. We selected these architectural paradigms for their broad generality and structural flexibility. Unlike convolution-based methods, which are fundamentally constrained by the requirement for uniform spatial grids, these selected models feature adaptable input-output mappings. This flexibility allows them to universally process highly complex, unstructured, and dimensionally diverse data representations. Given their capacity to handle such generalized inputs, we select GNOT (Hao et al., 2023), Transolver (Wu et al., 2024), Transolver++ (Luo et al.), LNO (Wang & Wang, 2024), and AMG (Li et al., 2024b) as our representative baselines.

Foundation models, pretrained on massive amounts of data, would theoretically serve as ideal baselines. However, current foundation models (Hao et al., 2024; McCabe et al., 2024; Ye et al., 2024; Herde et al., 2024; Shen et al., 2024; Zhou et al.; Wang et al., 2025) are predominantly restricted to grid-based domains. They lack exposure to the Eulerian mesh and Lagrangian particle representations that are central to our work. Furthermore, their rigid input-output architectures preclude them from generalizing across our highly unstructured data, which features diverse spatial dimensions and varied discretization schemes.

3.5 Multi-Agent Instantiation of Auto Research Pipeline

AutoPDEBench is agnostic to the internal design of $\mathcal{A}$. Any system that maps $\mathcal{I}_i$ to (m_i, ℓ_i, h_i) under the protocol above can be evaluated, including more elaborate multi-agent frameworks with dynamic delegation or parallel search. We provide a reference instantiation of $\mathcal{A}$ as a role-specialized multi-agent system of three LLM-driven components: a planner, a coder, and a debugger. The planner consumes the dataset description τ_{data}, the modeling objective τ_{obj}, the hardware constraints τ_{const}, the coding interface specification $\tau_{\text{interface}}$ (editable files and program structure), and the trial history τ_{history}, and produces a detailed step-by-step plan

$$\tau_{\text{plan}} = \mathcal{A}_{\text{planner}}\big(\tau_{\text{data}}, \tau_{\text{obj}}, \tau_{\text{const}}, \tau_{\text{interface}}, \tau_{\text{history}}\big), \tag{4}$$

a natural-language specification of the model architecture, loss functions, and hyperparameter choices to attempt this round, together with its rationale. The coder then translates this plan into executable files, conditioning on the hardware constraints, the coding interface specification, and the plan:

$$(m_i, \ell_i, h_i) = \mathcal{A}_{\text{coder}}\big(\tau_{\text{const}}, \tau_{\text{interface}}, \tau_{\text{plan}}\big). \tag{5}$$

We then attempt $\theta_i = \mathrm{Train}(\Phi_{m_i}, \mathcal{L}^{\text{train}}, h_i, \mathcal{D}_{\text{train}})$. If training fails, raising an execution error or violating the memory budget $\mathrm{Mem}(\Phi_{m_i}, h_i) \le B$, we invoke the debugger, which revises the files given the current code, the hardware constraints, the coding interface specification, the plan, and the observed error e:

$$(m_i, \ell_i, h_i) \leftarrow \mathcal{A}_{\text{debugger}}\big(m_i, \ell_i, h_i, \tau_{\text{const}}, \tau_{\text{interface}}, \tau_{\text{plan}}, e\big). \tag{6}$$

This step repeats until training completes successfully or a fixed retry budget is exhausted, in which case the round is recorded as a failure.

4 Experiments

We present the main results in Table 2, where we use RMSE as the primary evaluation metric and refer the readers to the Appendix of each dataset for additional metrics. Overall, we find that even this baseline multi-agent system achieves significantly better performance than general-purpose neural solvers. This underscores the value of tailoring architectures to address dataset-specific challenges and validates the efficacy of utilizing autonomous research agents for this task.

However, the agent struggles with highly complex datasets that remain open challenges in the literature. For instance, the **`LagFSI`** dataset models a sphere dropping into a body of water, with both phases discretized using Lagrangian particles. This scenario necessitates innovative solutions for two-way fluid-solid coupling, alongside methods to strictly enforce the rigid integrity of the sphere, which are non-trivial challenges that existing Lagrangian neural solvers have yet to successfully resolve. Consequently, the agent's generated design performs significantly worse than the

Dataset	GNOT	Transolver	Transolver++	LNO	AMG	Agent
HighSpeedFlow	1.826×10^{1}	3.232×10^{1}	3.271×10^{1}	4.482×10^{1}	2.819×10^{1}	$\mathbf{5.066\times10^{0}}$
RealData	5.638×10^{-3}	5.626×10^{-3}	5.618×10^{-3}	6.946×10^{-3}	4.889×10^{-3}	$\mathbf{4.458\times10^{-3}}$
MixedParam	3.417×10^{0}	4.359×10^{0}	4.329×10^{0}	4.346×10^{0}	3.128×10^{0}	$\mathbf{1.345\times10^{0}}$
Heterogeneous	4.207×10^{-3}	4.155×10^{-3}	3.963×10^{-3}	3.951×10^{-3}	3.900×10^{-3}	$\mathbf{3.311\times10^{-3}}$
Partial	1.978×10^{0}	3.636×10^{0}	3.636×10^{0}	3.636×10^{0}	2.111×10^{0}	$\mathbf{8.633\times10^{-2}}$
MixedDomain	1.784×10^{-3}	$\mathbf{1.739\times10^{-3}}$	1.775×10^{-3}	1.785×10^{-3}	6.491×10^{-2}	1.785×10^{-3}
MultiPhysics	6.902×10^{-2}	1.298×10^{-1}	1.345×10^{-1}	4.971×10^{-1}	7.303×10^{-2}	$\mathbf{5.141\times10^{-2}}$
3D	2.230×10^{-1}	2.279×10^{-1}	2.257×10^{-1}	2.238×10^{-1}	–	$\mathbf{7.182\times10^{-2}}$
IrregularTime	2.577×10^{0}	2.623×10^{0}	2.621×10^{0}	2.622×10^{0}	2.415×10^{0}	$\mathbf{6.511\times10^{-1}}$
LagInletOutlet	–	–	–	6.105×10^{-1}	–	$\mathbf{5.530\times10^{-1}}$
LagFSI	7.780×10^{-5}	$\mathbf{7.574\times10^{-5}}$	8.583×10^{-5}	8.455×10^{-5}	8.513×10^{-5}	2.215×10^{-4}
Extreme	–	2.289×10^{5}	2.190×10^{5}	4.813×10^{5}	–	$\mathbf{1.340\times10^{5}}$
MixedRes	3.629×10^{0}	3.616×10^{0}	3.564×10^{0}	3.481×10^{0}	–	$\mathbf{3.455\times10^{0}}$
ExtLag1	2.176×10^{-2}	3.526×10^{-2}	1.729×10^{-2}	1.407×10^{-2}	1.892×10^{-2}	$\mathbf{1.954\times10^{-3}}$
ExtLag2	1.365×10^{-3}	1.456×10^{-3}	1.551×10^{-3}	1.640×10^{-3}	1.566×10^{-3}	$\mathbf{5.483\times10^{-4}}$
MixedDim	3.615×10^{-2}	3.380×10^{-2}	3.357×10^{-2}	5.047×10^{-2}	–	$\mathbf{9.417\times10^{-3}}$
MixedBC	1.383×10^{-1}	8.500×10^{-2}	8.498×10^{-2}	8.511×10^{-2}	2.064×10^{-1}	$\mathbf{4.435\times10^{-2}}$
LongHorizon	5.674×10^{0}	4.787×10^{0}	4.756×10^{0}	$\mathbf{2.807\times10^{0}}$	7.088×10^{0}	1.781×10^{1}
Stochastic	3.989×10^{-3}	3.688×10^{-3}	3.171×10^{-3}	5.447×10^{-3}	3.442×10^{-3}	$\mathbf{2.324\times10^{-3}}$
MultiPhase	8.231×10^{-4}	9.163×10^{-4}	8.810×10^{-4}	8.203×10^{-4}	8.130×10^{-4}	$\mathbf{7.581\times10^{-4}}$
MixedPDE	8.792×10^{-2}	7.565×10^{-2}	7.535×10^{-2}	7.684×10^{-2}	8.845×10^{-2}	$\mathbf{2.701\times10^{-2}}$
LargeLag	1.300×10^{-2}	2.753×10^{-2}	2.027×10^{-2}	1.599×10^{-1}	–	$\mathbf{5.666\times10^{-3}}$
Sim2Real	1.987×10^{-3}	$\mathbf{1.783\times10^{-3}}$	1.809×10^{-3}	8.306×10^{-3}	2.705×10^{-3}	6.734×10^{-3}
HighDimensional	–	–	–	–	–	3.412×10^{-4}
ComplexGeometry	7.638×10^{-2}	8.295×10^{-2}	9.702×10^{-2}	1.032×10^{-1}	1.448×10^{-1}	1.366×10^{-1}

Table 2: Performance comparison of general-purpose neural solvers and agent-designed models. We report RMSE as the primary evaluation metric; additional metrics for each dataset are provided in the Appendix. Entries marked with a dash ("-") indicate that the model is either incompatible with the dataset or exceeds computational constraints (e.g., prohibitive training time). We use gpt-5.5 as the backbone.

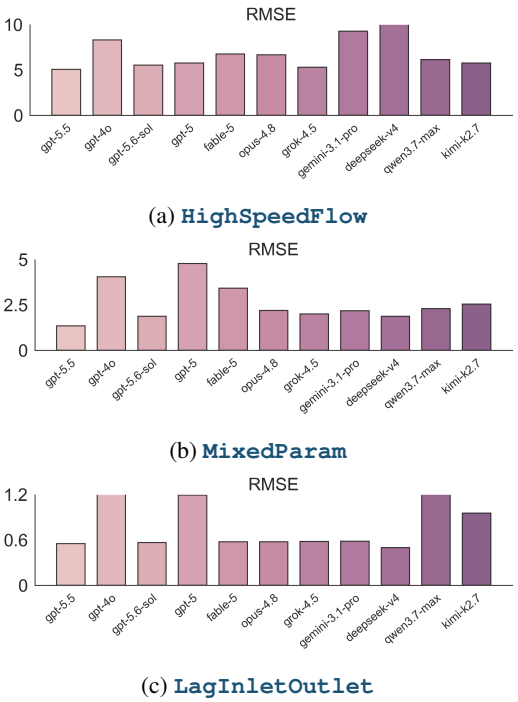
general-purpose baselines on this dataset. Another failure case is **LongHorizon**, which requires accurate autoregressive rollouts over 5,000 steps. While this horizon is substantially longer than those typically explored in existing research, it accurately reflects the strict demands of real-world engineering applications. Here again, the agent-designed model fails to surpass general-purpose baselines. We hypothesize that these shortcomings stem from the agent's exclusive reliance on its pre-trained knowledge to formulate solutions for entirely novel problems. Future work should therefore investigate mechanisms to dynamically integrate external domain expertise and latest scientific literature into the model design.

4.1 LLM Backbone Sensitivity Analysis

The primary results of our study are based on gpt-5.5 as the backbone for the agentic systems, as it was the frontier model at the time of this research. However, it is necessary to benchmark alternative LLM architectures. To achieve this, we replace the default gpt-5.5 backbone with a diverse suite of state-of-the-art LLMs, ensuring that the agentic framework, prompt templates, and search spaces remain strictly identical. For this experiment and all subsequent analyses, we evaluate on **HighSpeedFlow**, **LagInletOutlet**, and **MixedParam**, which serve as representative benchmarks for Eulerian data, Lagrangian particle data, and generalization tasks.

The comparative results are presented in Figure 3, with supplementary metrics detailed in Appendix F. Several flagship models—namely gpt-5.5, gpt-5.6-sol, fable-5, opus-4.8, and grok-4.5—consistently demonstrate strong performance across all datasets. In contrast, models such as deepseek-v4 and qwen-3.7-max exhibit high variance, excelling on certain tasks while underperforming on others. Meanwhile, older architectures like gpt-4o struggle significantly across these evaluations. We hypothesize that this inconsistent performance stems from limited exposure to these specific physical datasets during pre-training.

Notably, model performance on these physical tasks does not strictly align with general LLM leaderboards. For instance, while gpt-5.6-sol typically outperforms gpt-5.5 on broad intelligence benchmarks, it underperforms or merely matches gpt-5.5 across all three evaluated cases. This discrepancy indicates that general capabilities do not automatically transfer to these specialized domains, highlighting promising open research directions for targeted agentic training and the development of more robust agentic systems.



(a) **HighSpeedFlow**

(b) **MixedParam**

(c) **LagInletOutlet**

Figure 3: Performance comparison of various SOTA LLMs serving as the agent's backbone.

4.2 Comparison with Domain-Specific Baselines

In our primary evaluation, general-purpose neural solvers are benchmarked across the proposed datasets. While these general models represent current state-of-the-art and are capable of broad physical generalization, researchers have also engineered highly specialized, domain-specific architectures tailored to unique dataset characteristics. This experiment aims to contextualize our automated agent's performance against these specialized baselines.

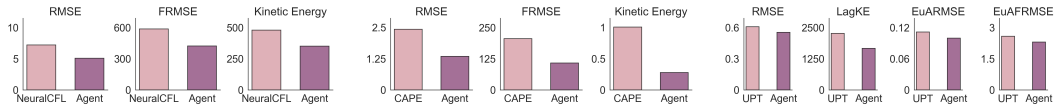
We selected a representative subset of experimental scenarios that feature well-documented, domain-specific baselines in the existing literature. An exhaustive comparison across all tasks is intractable due to the significant engineering overhead required to reproduce proprietary codebases, combined with the fact that several of our custom, challenge-driven datasets lack existing domain-specific counterparts. Instead, this evaluation serves as a targeted stress test to demonstrate whether the automated agent can match or exceed the performance of manually designed, domain-specific solvers.

We present the comparative results in Figure 4. For the **HighSpeedFlow** domain, we benchmark against NeuralCFL (Helwig et al., 2026), which introduces a two-phase deep learning architecture to dynamically adapt time steps. For **MixedParam**, we compare our method with CAPE (Takamoto et al., 2023), a model that leverages parameter-guided channel attention to improve generalization in PDE parameters. Finally, for **LagInletOutlet**, we evaluate against UPT (Alkin et al., 2024), a scalable transformer-based framework that demonstrates superior performance in Lagrangian field learning. The results demonstrate that the agent's design consistently outperforms these human-designed, dataset-specific baselines across all evaluated domains.

4.3 Transferability Across PDEs

The primary objective of our automated agent framework is to generate neural architectures that capture foundational mathematical and physical principles. Preliminary analyses indicate that these agent-designed architectures are transferable across PDEs. To empirically validate this observation, we evaluate the out-of-distribution transferability of the generated solvers. Specifically, architectures originally optimized by the agent for a target PDE are trained directly on entirely different PDEs that share similar dataset-specific challenges.

The results are presented in Figure 5. For **HighSpeedFlow**, we test on the coal dust explosion dataset from Helwig et al. (2026), which models the multiphase fluid dynamics of a normal shock wave sweeping over a layer of coal dust, triggering interfacial instabilities that evolve into turbulent vortices and drive gas-particle mixing. For **MixedParam**, we evaluate on the 2D Burgers' equation with kinematic viscosity varying between 5 and 50. Finally, for **LagInletOutlet**, we test on a 2D wave dataset where regular waves are generated at the left inlet using predefined velocities, and the right boundary features a slanted wall. The results demonstrate that the agent's designs



(a) **HighSpeedFlow** (b) **MixedParam** (c) **LagInletOutlet**

Figure 4: Performance comparison against domain-specific baselines. The automated agent's designs are evaluated against human engineered architectures across three representative datasets.



(a) **HighSpeedFlow** (b) **MixedParam** (c) **LagInletOutlet**

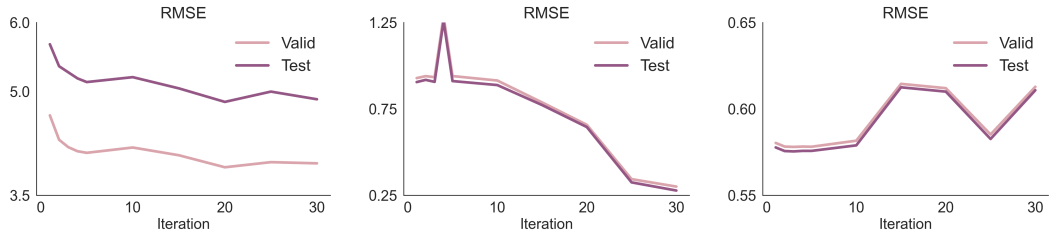
Figure 5: Transferability of agent-designed architectures across PDEs. Models originally optimized by the agent for specific target PDEs are evaluated on entirely different physical systems sharing similar data characteristics.

successfully transfer to these fundamentally distinct physical scenarios, validating that they effectively resolve dataset-specific issues without being structurally overfitted to their original equations. Additional metric evaluations are provided in Appendix E.

4.4 Agent Long-Horizon Scalability

Our base experimental pipeline limits the agent's optimization process to 5 iterations, which empirical testing identifies as a pragmatic threshold for achieving stable performance under reasonable resource constraints. This supplementary experiment extends the horizon to 30 iterations to evaluate long-term performance gains.

Results (Figure 6, Appendix I.18) show the agent successfully improves design for **HighSpeedFlow** and **MixedParam**, but struggles on **LagInletOutlet**. Notably, performance on **MixedParam** improves significantly after iteration 10 as the agent discovers superior architectures. Conversely, on **HighSpeedFlow** and **LagInletOutlet**, it relies on conservative hyperparameter tuning and minor tweaks. These findings suggest future agent designs must systematically identify model limitations.



(a) **HighSpeedFlow** (b) **MixedParam** (c) **LagInletOutlet**

Figure 6: Model performance across long-horizon optimization. The computational budget is extended to 30 iterations to evaluate long-term behavioral dynamics.

5 Conclusion

We introduce a comprehensive benchmark for evaluating LLM-driven automated research in neural PDE solvers. Our benchmark provides 25 challenging datasets featuring complex physical constraints. Our empirical evaluations demonstrate that an automated research agent can successfully design specialized solver pipelines that consistently outperform versatile baselines. We refer readers to Appendix A for broader impact and Appendix B for limitation and future work.

6 AI USE STATEMENT

In this work, we used generative AI tools for debugging code and for improving the clarity, flow, and grammar of our writing. We have reviewed all AI-assisted work. Specifically, all AI-suggested text revisions were manually reviewed by the authors to ensure they accurately preserved our original scientific meaning and intent, and all AI-assisted code debugging was rigorously tested and verified for correctness in our execution environment. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Shubham Agarwal, Gaurav Sahu, Abhay Puri, Issam H. Laradji, Krishnamurthy DJ Dvijotham, Jason Stanley, Laurent Charlin, and Christopher Pal. Litllm: A toolkit for scientific literature review, 2025. URL https://arxiv.org/abs/2402.01788.

Benedikt Alkin, Andreas Fürst, Simon Schmid, Lukas Gruber, Markus Holzleitner, and Johannes Brandstetter. Universal physics transformers. *arXiv preprint arXiv:2402.12365*, 2024.

Benedikt Alkin, Maurits Bleeker, Richard Kurle, Tobias Kronlachner, Reinhard Sonnleitner, Matthias Dorfer, and Johannes Brandstetter. AB-UPT: Scaling neural CFD surrogates for high- fidelity automotive aerodynamics simulations via anchored- branched universal physics transformers. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=nwQ8nitlTZ.

Oussama Boussif, Yoshua Bengio, Loubna Benabbou, and Dan Assouline. MAgnet: Mesh agnostic neural PDE solver. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=bx2roi8hca8.

Johannes Brandstetter, Daniel E. Worrall, and Max Welling. Message passing neural PDE solvers. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=vSix3HPYKSU.

Garyk Brixi, Matthew G. Durrant, Jerome Ku, Mohsen Naghipourfar, Michael Poli, Gwanggyu Sun, Greg Brockman, Daniel Chang, Alison Fanton, Gabriel A. Gonzalez, Samuel H. King, David B. Li, Aditi T. Merchant, Eric Nguyen, Chiara Ricci-Tam, David W. Romero, Jonathan C. Schmok, Ali Taghibakhshi, Anton Vorontsov, Brandon Yang, Myra Deng, Liv Gorton, Nam Nguyen, Nicholas K. Wang, Michael T. Pearce, Elana Simon, Etowah Adams, Zachary J. Amador, Euan A. Ashley, Stephen A. Baccus, Haoyu Dai, Steven Dillmann, Stefano Ermon, Daniel Guo, Michael H. Herschl, Rajesh Ilango, Ken Janik, Amy X. Lu, Reshma Mehta, Mohammad R. K. Mofrad, Madelena Y. Ng, Jaspreet Pannu, Christopher Ré, John St. John, Jeremy Sullivan, Joseph Tey, Ben Viggiano, Kevin Zhu, Greg Zynda, Daniel Balsam, Patrick Collison, Anthony B. Costa, Tina Hernandez-Boussard, Eric Ho, Ming-Yu Liu, Thomas McGrath, Kimberly Powell, Sudarshan Pinglay, Dave P. Burke, Hani Goodarzi, Patrick D. Hsu, and Brian L. Hie. Genome modelling and design across all domains of life with evo 2. *Nature*, 2026. doi: 10.1038/s41586-026-10176-5. URL https://doi.org/10.1038/s41586-026-10176-5.

Jun Shern Chan, Neil Chowdhury, Oliver Jaffe, James Aung, Dane Sherburn, Evan Mays, Giulio Starace, Kevin Liu, Leon Maksin, Tejal Patwardhan, Aleksander Madry, and Lilian Weng. MLE-bench: Evaluating machine learning agents on machine learning engineering. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=6s5uXNWGIh.

Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository, 2015. URL https://arxiv.org/abs/1512.03012.

Tianyu Chen, Haoyi Zhou, Ying Li, Hao Wang, Chonghan Gao, Rongye Shi, Shanghang Zhang, and Jianxin Li. Omniarch: Building foundation model for scientific computing. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=UlprLwWYKP.

Yimeng Chen, Piotr Piekos, Mateusz Ostaszewski, Firas Laakom, and Jürgen Schmidhuber. Physgym: Benchmarking LLMs in interactive physics discovery with controlled priors. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2026. URL https://openreview.net/forum?id=w8uII2qAmd.

J.M. Domínguez, G. Fourtakas, C. Altomare, R.B. Canelas, A. Tafuni, O. García-Feal, I. Martínez-Estévez, A. Mokos, R. Vacondio, A.J.C. Crespo, B.D. Rogers, P.K. Stansby, and M. Gómez-Gesteira. Dualsphysics: from fluid dynamics to multiphysics problems. *Computational Particle Mechanics*, 9(5):867–895, 2022. ISSN 2196-4386. doi: https://doi.org/10.1007/s40571-021-00404-2. URL https://www.sciencedirect.com/science/article/pii/S219643862500631X.

Huanshuo Dong, Keyao Zhang, Hong Wang, Zhezheng Hao, Zhiwei Zhuang, Ziyan Liu, Jiacong Wang, Gengyuan Liu, and Xin Jin. Autopde: Reliable agentic pde solving via explicitly represented solver strategies, 2026. URL https://arxiv.org/abs/2606.10752.

Fabio Ferreira, Lucca Wobbe, Arjun Krishnakumar, Frank Hutter, and Arber Zela. Can llms beat classical hyperparameter optimization algorithms? a study on autoresearch, 2026. URL https://arxiv.org/abs/2603.24647.

Muhan Gao, Jash Shah, Weiqi Wang, Kuan-Hao Huang, and Daniel Khashabi. Science hierarchography: Hierarchical organization of science literature, 2025. URL https://arxiv.org/abs/2504.13834.

Alireza Ghafarollahi and Markus J Buehler. Sciagents: Automating scientific discovery through bioinspired multi-agent intelligent graph reasoning. *Advanced Materials*, pp. 2413523, 2024.

Ali Essam Ghareeb, Benjamin Chang, Ludovico Mitchener, Angela Yiu, Caralyn J. Szostkiewicz, Jon M. Laurent, Muhammed T. Razzak, Andrew D. White, Michaela M. Hinks, and Samuel G. Rodriques. Robin: A multi-agent system for automating scientific discovery, 2025. URL https://arxiv.org/abs/2505.13400.

R. A. Gingold and J. J. Monaghan. Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Monthly Notices of the Royal Astronomical Society*, 181(3):375–389, 12 1977. ISSN 0035-8711. doi: 10.1093/mnras/181.3.375. URL https://doi.org/10.1093/mnras/181.3.375.

Gregory Go, Khang Ly, Anders Sogaard, Amin Reza Tabatabaei, Maarten de Rijke, and Xinyi Chen. Lira: A multi-agent framework for reliable and readable literature review generation. In *AAAI Conference on Artificial Intelligence*, 2025. URL https://api.semanticscholar.org/CorpusID:281886200.

Daniel Greenfeld, Meirav Galun, Ronen Basri, Irad Yavneh, and Ron Kimmel. Learning to optimize multigrid PDE solvers. In *International Conference on Machine Learning*, pp. 2415–2423. PMLR, 2019.

Taicheng Guo, Nitesh V. Chawla, Olaf Wiest, and Xiangliang Zhang. Autollmresearch: Training research agents for automating llm experiment configuration - learning from cheap, optimizing expensive, 2026. URL https://arxiv.org/abs/2605.11518.

Jayesh K Gupta and Johannes Brandstetter. Towards multi-spatiotemporal-scale generalized pde modeling. *arXiv preprint arXiv:2209.15616*, 2022.

Jiaqi Han, Wenbing Huang, Hengbo Ma, Jiachen Li, Joshua B Tenenbaum, and Chuang Gan. Learning physical dynamics with subequivariant graph neural networks. *arXiv preprint arXiv:2210.06876*, 2022.

Zhen Hang, Yushan Yashengjiang, Junhui Li, Huanshuo Dong, Yang Wei, Zhezheng Hao, Jiangtao Ma, Songlin Bai, Haozhong Kai, Xihang Yue, Gangzong Si, Dongming Jiang, Chao Yao, Zhanhua Hu, Jiangqing Zhang, Pengwei Liu, Yaomin Shen, Xingyu Ren, Lei Liu, Zikang Xu, Han Li, Qingsong Yao, Hande Dong, and Hong Wang. Pdeagent-bench: A multi-metric, multi-library benchmark for pde solver generation, 2026. URL https://arxiv.org/abs/2605.09636.

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. *ArXiv*, abs/2302.14376, 2023. URL https://api.semanticscholar.org/CorpusID:257232579.

Zhongkai Hao, Chang Su, Songming Liu, Julius Berner, Chengyang Ying, Hang Su, Anima Anandkumar, Jian Song, and Jun Zhu. Dpot: Auto-regressive denoising operator transformer for large-scale pde pre-training. *arXiv preprint arXiv:2403.03542*, 2024.

Jacob Helwig, Sai Sreeharsha Adavi, Xuan Zhang, Yuchao Lin, Felix S. Chim, Luke Takeshi Vizzini, Haiyang Yu, Muhammad Hasnain, Saykat Kumar Biswas, John J. Holloway, Narendra Singh, N. K. Anand, Swagnik Guhathakurta, and Shuiwang Ji. A two-phase deep learning framework for adaptive time-stepping in high-speed flow modeling, 2026. URL https://arxiv.org/abs/2506.07969.

Maximilian Herde, Bogdan Raonic, Tobias Rohner, Roger Käppeli, Roberto Molinaro, Emmanuel de Bezenac, and Siddhartha Mishra. Poseidon: Efficient foundation models for PDEs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=JC1VKK3UXk.

Masanobu Horie and NAOTO MITSUME. Physics-embedded neural networks: Graph neural PDE solvers with mixed boundary conditions. In *NeurIPS 2022 AI for Science: Progress and Promises*, 2022. URL https://openreview.net/forum?id=gyIRLZPGiG.

Peiyan Hu, Haodong Feng, Hongyuan Liu, Tongtong Yan, Wenhao Deng, Tianrun Gao, Rong Zheng, Haoren Zheng, Chenglei Yu, Chuanrui Wang, Kaiwen Li, Zhi-Ming Ma, Dezhi Zhou, Xingcai Lu, Dixia Fan, and Tailin Wu. Realpdebench: A benchmark for complex physical systems with real-world data. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=y3oHMcoItR. Oral Presentation.

Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. Mlagentbench: Evaluating language agents on machine learning experimentation. In *International Conference on Machine Learning*, 2023. URL https://api.semanticscholar.org/CorpusID:263671541.

Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R Narasimhan. SWE-bench: Can language models resolve real-world github issues? In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=VTF8yNQM66.

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

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to pdes. *JMLR*, 24(1), 2023.

Ruochen Li, Teerth Patel, Qingyun Wang, Qingyun Wang, and Xinya Du. Mlr-copilot: Autonomous machine learning research based on large language models agents. *ArXiv*, abs/2408.14033, 2024a. URL https://api.semanticscholar.org/CorpusID:271957477.

Ruoyan Li, Zijie Huang, Haixin Wang, Guancheng Wan, Yizhou Sun, and Wei Wang. Self-guided diffusion model for accelerating computational fluid dynamics, 2025a. URL https://arxiv.org/abs/2504.04375.

Ruoyan Li, Guancheng Wan, Zijie Huang, Zixiao Liu, Haixin Wang, Xiao Luo, Wei Wang, and Yizhou Sun. Flow field reconstruction with sensor placement policy learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=1esFEjMUBS.

Shanda Li, Tanya Marwah, Junhong Shen, Weiwei Sun, Andrej Risteski, Yiming Yang, and Ameet Talwalkar. Codepde: An inference framework for llm-driven pde solver generation. *arXiv preprint arXiv:2505.08783*, 2025c.

Zhihao Li, Haoze Song, Di Xiao, Zhilu Lai, and Wei Wang. Harnessing scale and physics: A multi-graph neural operator framework for pdes on arbitrary geometries. *arXiv preprint arXiv:2411.15178*, 2024b.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Multipole graph neural operator for parametric partial differential equations, 2020. URL https://arxiv.org/abs/2006.09535.

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

Zongyi Li, 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 *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=86dXbqT5Ua.

Mario Lino, Chris Cantwell, Anil A. Bharath, and Stathi Fotiadis. Simulating continuum mechanics with multi-scale graph neural networks, 2021. URL https://arxiv.org/abs/2106.04900.

B. List, L.-W. Chen, and N. Thuerey. Learned turbulence modelling with differentiable fluid solvers: physics-based loss functions and optimisation horizons. *Journal of Fluid Mechanics*, 949:A25, 2022.

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.

Huakun Luo, Haixu Wu, Hang Zhou, Lanxiang Xing, Yichen Di, Jianmin Wang, and Mingsheng Long. Transolver++: An accurate neural solver for pdes on million-scale geometries. In *Forty-second International Conference on Machine Learning*.

Michael McCabe, Bruno Régaldo-Saint Blancard, Liam Holden Parker, Ruben Ohana, Miles Cranmer, Alberto Bietti, Michael Eickenberg, Siavash Golkar, Geraud Krawezik, Francois Lanusse, Mariel Pettee, Tiberiu Tesileanu, Kyunghyun Cho, and Shirley Ho. Multiple physics pretraining for spatiotemporal surrogate models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=DKSI3bULiZ.

Matteo Merler, Katsiaryna Haitsiukevich, Nicola Dainese, and Pekka Marttinen. In-context symbolic regression: Leveraging large language models for function discovery. In Xiyan Fu and Eve Fleisig (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pp. 589–606, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-srw.49. URL https://aclanthology.org/2024.acl-srw.49.

Ludovico Mitchener, Angela Yiu, Benjamin Chang, Mathieu Bourdenx, Tyler Nadolski, Arvis Sulovari, Eric C. Landsness, Daniel L. Barabasi, Siddharth Narayanan, Nicky Evans, Shriya Reddy, Martha Foiani, Aizad Kamal, Leah P. Shriver, Fang Cao, Asmamaw T. Wassie, Jon M. Laurent, Edwin Melville-Green, Mayk Caldas, Albert Bou, Kaleigh F. Roberts, Sladjana Zagorac, Timothy C. Orr, Miranda E. Orr, Kevin J. Zwezdaryk, Ali E. Ghareeb, Laurie McCoy, Bruna Gomes, Euan A. Ashley, Karen E. Duff, Tonio Buonassisi, Tom Rainforth, Randall J. Bateman, Michael Skarlinski, Samuel G. Rodriques, Michaela M. Hinks, and Andrew D. White. Kosmos: An ai scientist for autonomous discovery, 2025. URL https://arxiv.org/abs/2511.02824.

Roberto Molinaro, Yunan Yang, Björn Engquist, and Siddhartha Mishra. Neural inverse operators for solving pde inverse problems, 2023. URL https://arxiv.org/abs/2301.11167.

Sepehr Mousavi, Shizheng Wen, Levi Lingsch, Maximilian Herde, Bogdan Raonić, and Siddhartha Mishra. Rigno: A graph-based framework for robust and accurate operator learning for pdes on arbitrary domains. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, M. Pawan Kumar, Abigail See, Swarat Chaudhuri, George Holland, Alex Davies, Sebastian Nowozin, Pushmeet Kohli, and Matej Balog. Alphaevolve: A coding agent for scientific and algorithmic discovery, 2025. URL https://arxiv.org/abs/2506.13131.

Ruben Ohana, Michael McCabe, Lucas Meyer, Rudy Morel, Fruzsina Agocs, Miguel Beneitez, Marsha Berger, Blakesly Burkhart, Stuart Dalziel, Drummond Fielding, et al. The well: a large-scale collection of diverse physics simulations for machine learning. *Advances in Neural Information Processing Systems*, 37:44989–45037, 2024.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019. URL https://arxiv.org/abs/1912.01703.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter Battaglia. Learning mesh-based simulation with graph networks. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=roNqYL0_XP.

Lukas Prantl, Benjamin Ummenhofer, Vladlen Koltun, and Nils Thuerey. Guaranteed conservation of momentum for learning particle-based fluid dynamics. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=6niwHlzh10U.

Yansheng Qiu, Haoquan Zhang, Zhaopan Xu, Ming Li, Diping Song, Zheng Wang, and Kaipeng Zhang. Ai idea bench 2025: Ai research idea generation benchmark, 2025. URL https://arxiv.org/abs/2504.14191.

Md Ashiqur Rahman, Zachary E Ross, and Kamyar Azizzadenesheli. U-NO: U-shaped neural operators. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=j3oQF9coJd.

Prabhu Ramachandran, Aditya Bhosale, Kunal Puri, Pawan Negi, Abhinav Muta, A. Dinesh, Dileep Menon, Rahul Govind, Suraj Sanka, Amal S. Sebastian, Ananyo Sen, Rohan Kaushik, Anshuman Kumar, Vikas Kurapati, Mrinalgouda Patil, Deep Tavker, Pankaj Pandey, Chandrashekhar Kaushik, Arkopal Dutt, and Arpit Agarwal. PySPH: A Python-based Framework for Smoothed Particle Hydrodynamics. *ACM Transactions on Mathematical Software*, 47(4):1–38, December 2021. ISSN 0098-3500, 1557-7295. doi: 10.1145/3460773.

Omer Rochman-Sharabi, Sacha Lewin, and Gilles Louppe. A neural material point method for particle-based emulation. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=zSK81A2hxQ.

Nadim Saad, Gaurav Gupta, Shima Alizadeh, and Danielle C. Maddix. Guiding continuous operator learning through physics-based boundary constraints. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=gfWNItGOES6.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W. Battaglia. Learning to simulate complex physics with graph networks, 2020. URL https://arxiv.org/abs/2002.09405.

J. Sappl, L. Seiler, M. Harders, and W. Rauch. Deep learning of preconditioners for conjugate gradient solvers in urban water related problems. *arXiv preprint*, 2019.

Samuel Schmidgall, Yusheng Su, Ze Wang, Ximeng Sun, Jialian Wu, Xiaodong Yu, Jiang Liu, Michael Moor, Zicheng Liu, and Emad Barsoum. Agent laboratory: Using llm agents as research assistants, 2025. URL https://arxiv.org/abs/2501.04227.

Junhong Shen, Tanya Marwah, and Ameet Talwalkar. UPS: Efficiently building foundation models for PDE solving via cross-modal adaptation. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=0r9mhjRv1E.

Michael D. Skarlinski, Sam Cox, Jon M. Laurent, James D. Braza, Michaela Hinks, Michael J. Hammerling, Manvitha Ponnapati, Samuel G. Rodriques, and Andrew D. White. Language agents achieve superhuman synthesis of scientific knowledge, 2024. URL https://arxiv.org/abs/2409.13740.

Zeyuan Song, Xiaocong Zhen, and Zheyu Jiang. Can large language models design effective neural operators for solving partial differential equations? In *3rd AI for Math Workshop: Toward Self-Evolving Scientific Agents*, 2026. URL https://openreview.net/forum?id=6GCIuS0oO0.

Z. Sun, Y. Yang, and S. Yoo. A neural PDE solver with temporal stencil modeling. *arXiv preprint*, 2023.

Makoto Takamoto, Francesco Alesiani, and Mathias Niepert. Learning neural pde solvers with parameter-guided channel attention, 2023. URL https://arxiv.org/abs/2304.14118.

Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Dan MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. Pdebench: An extensive benchmark for scientific machine learning, 2024. URL https://arxiv.org/abs/2210.07182.

Jiabin Tang, Lianghao Xia, Zhonghang Li, and Chao Huang. Ai-researcher: Autonomous scientific innovation, 2025. URL https://arxiv.org/abs/2505.18705.

Artur Toshev, Gianluca Galletti, Fabian Fritz, Stefan Adami, and Nikolaus Adams. Lagrangebench: A lagrangian fluid mechanics benchmarking suite. *Advances in Neural Information Processing Systems*, 36, 2024a.

Artur P. Toshev, Gianluca Galletti, Johannes Brandstetter, Stefan Adami, and Nikolaus A. Adams. Learning lagrangian fluid mechanics with e(3)-equivariant graph neural networks, 2023.

Artur P. Toshev, Jonas A. Erbesdobler, Nikolaus A. Adams, and Johannes Brandstetter. Neural sph: Improved neural modeling of lagrangian fluid dynamics, 2024b. URL https://arxiv.org/abs/2402.06275.

Alasdair Tran, Alexander Mathews, Lexing Xie, and Cheng Soon Ong. Factorized fourier neural operators. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=tmIiMPl4IPa.

Tapas Tripura and Souvik Chakraborty. Wavelet neural operator for solving parametric partial differential equations in computational mechanics problems. *Computer Methods in Applied Mechanics and Engineering*, 404:115783, 2023.

Nobuyuki Umetani and Bernd Bickel. Learning three-dimensional flow for interactive aerodynamic design. *ACM Trans. Graph.*, 37(4), July 2018. ISSN 0730-0301. doi: 10.1145/3197517.3201325. URL https://doi.org/10.1145/3197517.3201325.

Haixin Wang, LI Jiaxin, Anubhav Dwivedi, Kentaro Hara, and Tailin Wu. BENO: Boundary-embedded neural operators for elliptic PDEs. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=ZZTkLDRmkg.

Haixin Wang, Ruoyan Li, Fred Xu, Fang Sun, Kaiqiao Han, Zijie Huang, Ching Chang, Xiao Luo, Wei Wang, and Yizhou Sun. Fd-bench: A modular and fair benchmark for data-driven fluid simulation, 2026a. URL https://arxiv.org/abs/2505.20349.

Hong Wang, Haiyang Xin, Jie Wang, Xuanze Yang, Fei Zha, Huanshuo Dong, and Yan Jiang. Mixture-of-experts operator transformer for large-scale pde pre-training, 2025. URL https://arxiv.org/abs/2510.25803.

Tian Wang and Chuang Wang. Latent neural operator for solving forward and inverse pde problems. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Xiaoxuan Wang, Han Zhang, Haixin Wang, Yidan Shi, Ruoyan Li, Kaiqiao Han, Chenyi Tong, Haoran Deng, Alexander K Taylor, Renliang Sun, Yanqiao Zhu, Jason Cong, Yizhou Sun, and Wei Wang. ARLArena: A unified framework for stable agentic reinforcement learning. In *Forty-third International Conference on Machine Learning*, 2026b. URL https://openreview.net/forum?id=90kxFi9VGP.

Shizheng Wen, Arsh Kumbhat, Levi Lingsch, Sepehr Mousavi, Yizhou Zhao, Praveen Chandrashekar, and Siddhartha Mishra. Geometry aware operator transformer as an efficient and accurate neural surrogate for pdes on arbitrary domains. 2025.

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

Haoyang Wu, Xinxin Zhang, and Lailai Zhu. Automated code development for pde solvers using large language models, 2025. URL https://arxiv.org/abs/2509.25194.

Qingpo Wuwu, Chonghan Gao, Tianyu Chen, Yihang Huang, Yuekai Zhang, Jianing Wang, Jianxin Li, Haoyi Zhou, and Shanghang Zhang. PINNsagent: Automated PDE surrogation with large language models. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=RO5OGOzs6M.

Zonglin Yang, Wanhao Liu, Ben Gao, Tong Xie, Yuqiang Li, Wanli Ouyang, Soujanya Poria, Erik Cambria, and Dongzhan Zhou. MOOSE-chem: Large language models for rediscovering unseen chemistry scientific hypotheses. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=X9OfMNNepI.

Zhanhong Ye, Xiang Huang, Leheng Chen, Hongsheng Liu, Zidong Wang, and Bin Dong. PDEformer: Towards a foundation model for one-dimensional partial differential equations. In *ICLR 2024 Workshop on AI4DifferentialEquations In Science*, 2024. URL https://openreview.net/forum?id=GLDMCwdhTK.

Yi Zhan, Min Luo, and Abbas Khayyer. Dualsphysics+: An enhanced dualsphysics with improvements in accuracy, energy conservation and resolution of the continuity equation. *Computer Physics Communications*, 306:109389, 2025. ISSN 0010-4655. doi: https://doi.org/10.1016/j.cpc.2024.109389. URL https://www.sciencedirect.com/science/article/pii/S0010465524003126.

Tao Zhang, Zhenhai Liu, Feipeng Qi, Yongjun Jiao, and Tailin Wu. M2pde: Compositional generative multiphysics and multi-component pde simulation, 2025. URL https://arxiv.org/abs/2412.04134.

Weiqun Zhang, Ann Almgren, Vince Beckner, John Bell, Johannes Blaschke, Cy Chan, Marcus Day, Brian Friesen, Kevin Gott, Daniel Graves, Max P. Katz, Andrew Myers, Tan Nguyen, Andrew Nonaka, Michele Rosso, Samuel Williams, and Michael Zingale. Amrex: a framework for block-structured adaptive mesh refinement. *Journal of Open Source Software*, 4(37):1370, 2019. doi: 10.21105/joss.01370. URL https://doi.org/10.21105/joss.01370.

Hongkai Zheng, Wenda Chu, Bingliang Zhang, Zihui Wu, Austin Wang, Berthy Feng, Caifeng Zou, Yu Sun, Nikola Borislavov Kovachki, Zachary E Ross, Katherine Bouman, and Yisong Yue. Inversebench: Benchmarking plug-and-play diffusion priors for inverse problems in physical sciences. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=U3PBITXNG6.

Hang Zhou, Yuezhou Ma, Haixu Wu, Haowen Wang, and Mingsheng Long. Unisolver: Pde-conditional transformers towards universal neural pde solvers. In *Forty-second International Conference on Machine Learning*.

A Broader Impact

Beyond the immediate advancements in neural PDE modeling, our work serves as a controlled, rigorous platform for investigating the capabilities of autonomous research agents in complex scientific domains. By focusing specifically on partial differential equations, the mathematical bedrock for critical applications such as weather and climate forecasting, aerospace engineering, fluid dynamics, and materials science, our benchmark grounds agentic AI evaluation in high-stakes, real-world physical challenges. We hope that by demonstrating the viability of AI-driven pipeline design, this platform will help pave the way for future autonomous research capable of accelerating scientific discovery across multiple disciplines. Finally, we do not identify any negative societal impacts associated with this work.

B Limitation and Future Work

A potential criticism of our work is that the number of samples is relatively small compared to other LLM benchmarks (Wang et al., 2026b; Jimenez et al., 2024). However, we emphasize that our benchmark is fundamentally different from traditional evaluation sets constructed by scraping vast amounts of existing text or code from the internet. Instead, our datasets are meticulously curated by domain experts who have identified these specific physical scenarios as either actively studied bottlenecks or entirely novel open problems. Because these challenges are highly specialized and recent, LLMs have no prior exposure to their solutions in their training corpora. Consequently, any successful modeling pipeline generated by an agent on these tasks represents significant, genuine progress in the field of neural PDE solvers, rather than mere data memorization.

We intend to maintain our work as a dynamic, rolling benchmark. We will progressively incorporate new datasets as novel challenges emerge in the field, and we actively encourage the broader research community to contribute their own complex datasets to our platform. In future work, we aim to collaborate with scientists across a wider spectrum of disciplines to identify and integrate even more challenging physical systems, specifically those characterized by idiosyncratic constraints that currently cause existing general-purpose solvers to fail. By continuously expanding this repository, we hope to further stress-test and advance the capabilities of autonomous AI research agents in scientific discovery.

C Evaluation Metrics

All metrics have the form $\mathcal{L}(\Phi_\theta; \mathcal{D})$. They compare the prediction $\hat{y} = \Phi_\theta(x, c)$ against the ground-truth target state y. We write $N = |\mathcal{D}|$ for the number of evaluated samples and index them by n. The target and prediction share the same representation, which takes one of three forms depending on the dataset. An Eulerian grid sample is $y \in \mathbb{R}^{C\times H\times W}$, storing C channels on a structured $H \times W$ grid whose nodes are fixed in space, with entries $y_{n,c,i,j}$ and predictions $\hat{y}_{n,c,i,j}$. An Eulerian mesh sample is $y \in \mathbb{R}^{M\times C}$, storing C channels on M fixed, unstructured mesh nodes indexed by m, giving $y_{n,m,c}$. A Lagrangian particle sample is $y \in \mathbb{R}^{M\times d}$, storing the M particle positions that move with the flow, where d is the spatial dimension; particle i has position $y_{n,i} \in \mathbb{R}^d$. Particle velocities $v_{n,i}$ are obtained from the positions by finite differencing. We define these metrics in two dimensions and they generalize naturally to higher-dimensional spaces.

Root Mean Squared Error (RMSE). The pointwise root mean squared error measures the mean per-node discrepancy between the predicted and target states, averaged over all samples and over every channel and spatial location. Its summation domain follows the state representation. For an Eulerian grid,

$$\mathrm{RMSE}(\Phi_\theta; \mathcal{D}) = \sqrt{\frac{1}{NCHW} \sum_{n,c,i,j} \left(\hat{y}_{n,c,i,j} - y_{n,c,i,j}\right)^2}. \tag{7}$$

For an Eulerian mesh and Lagrangian particles, the sum runs over the M mesh nodes,

$$\mathrm{RMSE}(\Phi_\theta; \mathcal{D}) = \sqrt{\frac{1}{NMC} \sum_{n,m,c} \left(\hat{y}_{n,m,c} - y_{n,m,c}\right)^2}. \tag{8}$$

Fourier Root Mean Squared Error (FRMSE). The Fourier RMSE compares the amplitude spectra of Eulerian-grid states, penalizing errors in how energy is distributed across spatial frequencies rather than in physical space. This is helpful because two fields can have a similar pointwise RMSE yet differ sharply in their spectral content. A model may capture the large-scale structure while smearing out fine features, or introduce spurious high-frequency noise. By comparing magnitude spectra, FRMSE exposes such discrepancies and rewards models that reproduce the correct multi-scale behavior of the flow. Let $\hat{Y} = \mathcal{F}[\hat{y}]$ and $Y = \mathcal{F}[y]$ be the discrete Fourier transforms over the spatial dimensions, with frequency indices (k, ℓ):

$$\mathrm{FRMSE}(\Phi_\theta; \mathcal{D}) = \sqrt{\frac{1}{NCHW} \sum_{n,c,k,\ell} \Big(\big|\hat{Y}_{n,c,k,\ell}\big| - \big|Y_{n,c,k,\ell}\big| \Big)^2}. \tag{9}$$

Kinetic Energy Error (KE). With velocity components on channels (c_u, c_v), the pointwise kinetic energy density is $k_{n,i,j} = \frac{1}{2}\big(y^2_{n,c_u,i,j} + y^2_{n,c_v,i,j}\big)$. This metric reports the mean absolute error of that density, isolating how faithfully the model reproduces the flow's energy content, a physically meaningful quantity that pointwise state error does not directly measure. It is helpful for detecting systematic bias in the velocity magnitude, such as a model that is over- or under-energetic, which can signal unstable or overly diffusive dynamics even when the velocity fields appear qualitatively reasonable. The metric is

$$E_{\mathrm{KE}}(\Phi_\theta; \mathcal{D}) = \frac{1}{NHW} \sum_{n,i,j} \big|\hat{k}_{n,i,j} - k_{n,i,j}\big|. \tag{10}$$

Magnetic Energy Error (MagE). Analogously to the kinetic energy error, with magnetic-field components on channels (b_x, b_y) the pointwise magnetic energy density is $m_{n,i,j} = \frac{1}{2}\big(y^2_{n,b_x,i,j} + y^2_{n,b_y,i,j}\big)$. This metric reports the mean absolute error of that density, capturing how well the model reproduces the energy stored in the magnetic field, which governs the dynamics of magnetohydrodynamic systems. It is helpful for revealing whether a model conserves or spuriously injects magnetic energy, a failure mode that pointwise field error can mask but that strongly affects the physical fidelity of the predicted evolution. The metric is

$$E_{\mathrm{ME}}(\Phi_\theta; \mathcal{D}) = \frac{1}{NHW} \sum_{n,i,j} \big|\hat{m}_{n,i,j} - m_{n,i,j}\big|. \tag{11}$$

Mass Error (MassE). Specific to the advection equation, where total mass is the physically conserved quantity. The total mass of a state is its discrete spatial integral $M_{n,c} = \sum_{i,j} y_{n,c,i,j}$, and the metric is the RMSE of this quantity, reduced over all samples and channels. This is helpful because it directly probes a known conservation law: a faithful simulator should neither create nor destroy mass over time, so any deviation exposes non-physical drift that a purely pointwise metric would not single out. The metric is

$$\mathrm{RMSE}_{\mathrm{mass}}(\Phi_\theta; \mathcal{D}) = \sqrt{\frac{1}{NC} \sum_{n,c} \big(\hat{M}_{n,c} - M_{n,c}\big)^2}. \tag{12}$$

Boundary RMSE (BoundaryRMSE). The RMSE restricted to the outer frame of the domain $\partial\Omega = \{(i,j) : i \in \{0, H\} \text{ or } j \in \{0, W\}\}$, isolating error in the enforcement of the boundary conditions. This is helpful because boundary behavior is often critical to a simulation's correctness yet contributes only a small fraction of the nodes, so its error is easily diluted in a whole-domain average. By evaluating only the boundary, the metric surfaces whether a model respects the prescribed conditions rather than trading boundary accuracy for lower interior error. The metric is

$$\mathrm{RMSE}_{\partial\Omega}(\Phi_\theta; \mathcal{D}) = \sqrt{\frac{1}{NC\,|\partial\Omega|} \sum_{n,c} \sum_{(i,j)\in\partial\Omega} \big(\hat{y}_{n,c,i,j} - y_{n,c,i,j}\big)^2}. \tag{13}$$

Lagrangian Kinetic Energy Error (LagKE). Evaluated directly on particles with unit mass $m_i = 1$. The per-sample kinetic energy is

$$\mathrm{KE}_n = \sum_{i=1}^{N_p} \|\mathbf{v}_{n,i}\|^2, \tag{14}$$

and the metric is the RMSE of this scalar across the split. This is helpful because it summarizes the total energy of the particle system in a single physically meaningful quantity, exposing whether a model produces an over- or under-energetic flow. Since the velocities $\mathbf{v}_{n,i}$ are finite-differenced from predicted positions, the metric is also sensitive to temporally inconsistent that may nonetheless have plausible instantaneous positions. The metric is

$$\mathrm{RMSE}_{\mathrm{KE}}(\Phi_\theta;\mathcal{D}) = \sqrt{\frac{1}{N}\sum_n \big(\hat{\mathrm{KE}}_n - \mathrm{KE}_n\big)^2}. \tag{15}$$

Eulerian-Aggregated Lagrangian Error (EuARMSE, EuAFRMSE). To evaluate Lagrangian particle states with the grid metrics above, particle velocities are first interpolated onto cell-centered Eulerian grid nodes $\mathbf{x}_g$ using an SPH kernel. Writing $W(\cdot)$ for the kernel, the interpolated node velocity is

$$\tilde{\mathbf{u}}(\mathbf{x}_g) = \frac{\sum_j \mathbf{v}_j\, W(\|\mathbf{x}_g - \mathbf{r}_j\|)}{\sum_j W(\|\mathbf{x}_g - \mathbf{r}_j\|)}. \tag{16}$$

The grid RMSE and FRMSE are then applied to the resulting predicted and target node fields, yielding the Eulerian-aggregated RMSE (EuARMSE) and Eulerian-aggregated FRMSE (EuAFRMSE). This is helpful because it measures the practical utility of a predicted particle trajectory. Querying a physical quantity from a Lagrangian state at fixed spatial locations requires exactly this kernel aggregation, so the metric evaluates the particles as they would actually be consumed downstream.

Lagrangian Sinkhorn Divergence (Sinkhorn). An entropic optimal-transport metric on the particle position sets, comparing the predicted point cloud $\hat{P} = \{\hat{y}_i\}$ against the target $P = \{y_i\}$. This is helpful because it compares the two configurations as unordered distributions rather than by particle-wise correspondence, which is the natural notion of error when particles are interchangeable or when their indexing carries no physical meaning. It rewards a model that places particles in the right spatial distribution even if individual identities are permuted, capturing bulk structural agreement that a per-particle position error would penalize spuriously. With uniform weights $a = b = \frac{1}{N_p}\mathbf{1}$, squared-Euclidean cost $M_{ij} = \|y_i - y_j\|^2$, and entropic regularization ε, let $\mathrm{OT}_\varepsilon(P, Q)$ denote the Sinkhorn transport cost. The divergence debiases the raw cost with the self-transport terms:

$$S_\varepsilon(\Phi_\theta;\mathcal{D}) = \max\Big(\mathrm{OT}_\varepsilon(\hat{P}, P) - \tfrac{1}{2}\big[\mathrm{OT}_\varepsilon(\hat{P}, \hat{P}) + \mathrm{OT}_\varepsilon(P, P)\big],\ 0\Big). \tag{17}$$

Coefficient Error (Coeff). For each sample, a scalar physical coefficient κ (e.g., a drag or lift coefficient) is derived from the predicted state. The metric is its RMSE against the reference value across the split. This is helpful because such coefficients are the integrated, task-level quantities that practitioners ultimately care about, and reducing an entire field to a single engineering-relevant scalar tests whether the prediction is accurate where it matters for the downstream application. A model can attain low pointwise error yet still misestimate the coefficient if it misplaces the local features that dominate the integral, which this metric is designed to expose. The metric is

$$\mathrm{RMSE}_\kappa(\Phi_\theta;\mathcal{D}) = \sqrt{\frac{1}{N}\sum_n \big(\hat{\kappa}_n - \kappa_n\big)^2}. \tag{18}$$

D Hardware Specifications

We implement models in PyTorch (Paszke et al., 2019). All experiments can be run on servers/-workstations with the following configuration:

- 80 CPUs, 503G Mem, 8 x NVIDIA V100 GPUs.
- 48 CPUs, 220G Mem, 8 x NVIDIA TITAN XP GPUs.
- 96 CPUs, 1.0T Mem, 8 x NVIDIA A100 GPUs.
- 64 CPUs, 1.0T Mem, 8 x NVIDIA RTX A6000 GPUs.
- 224 CPUs, 1.5T Mem, 8 x NVIDIA L40S GPUs.
- 128 CPUs, 1.5T Mem, 8 x NVIDIA H200 GPUs.
- 128 CPUs, 480G Mem, 8 × NVIDIA RTX 4090 GPUs.

E SUPPLEMENTARY EXPERIMENTS ON TRANSFERABILITY ACROSS PDES

We present supplementary experimental results regarding transferability across PDEs in Tables 3, 4, and 5, including FRMSE and Kinetic Energy as additional evaluation metrics for Eulerian data and LagKE, EuARMSE, and EuAFRMSE for Lagrangian data. Specifically, we transfer the agent's design from **`HighSpeedFlow`** to a coal dust explosion dataset, from **`MixedParam`** to the 2D Burgers equation, and from **`LagInletOutlet`** to a Lagrangian inlet wave on a slanted wall dataset. We observe that the agent's designs successfully transfer to these new environments while maintaining optimal performance.

	RMSE	FRMSE	Kinetic Energy
GNOT	3.442×10^{0}	4.985×10^{2}	3.557×10^{2}
Transolver	1.182×10^{1}	1.786×10^{3}	5.664×10^{2}
Transolver++	1.172×10^{1}	1.780×10^{3}	5.655×10^{2}
LNO	2.439×10^{1}	4.155×10^{3}	5.585×10^{3}
AMG	5.930×10^{0}	8.538×10^{2}	4.740×10^{2}
Agent	$\mathbf{1.391 \times 10^{0}}$	$\mathbf{1.890 \times 10^{2}}$	$\mathbf{5.134 \times 10^{1}}$

Table 3: Transferability of **`HighSpeedFlow`** to coal dust explosion dynamics. We evaluate the agent-optimized architecture in an out-of-distribution setting on a novel coal dust explosion dataset. The agent-designed solver maintains superior accuracy across RMSE, FRMSE, and Kinetic Energy metrics compared to general-purpose baselines.

	RMSE	FRMSE	Kinetic Energy
GNOT	1.531×10^{-2}	1.954×10^{0}	1.426×10^{-4}
Transolver	1.531×10^{-2}	1.953×10^{0}	1.424×10^{-4}
Transolver++	1.531×10^{-2}	1.954×10^{0}	1.425×10^{-4}
LNO	1.856×10^{-2}	2.373×10^{0}	1.774×10^{-4}
AMG	1.097×10^{-2}	1.306×10^{0}	9.326×10^{-5}
Agent	$\mathbf{7.718 \times 10^{-3}}$	$\mathbf{9.720 \times 10^{-1}}$	$\mathbf{9.125 \times 10^{-5}}$

Table 4: Transferability of **`MixedParam`** to the 2D Burgers' equation. We evaluate the agent-optimized architecture in an out-of-distribution setting on the 2D Burgers' equation dataset. The agent-designed solver maintains superior accuracy across RMSE, FRMSE, and Kinetic Energy metrics compared to general-purpose baselines.

	RMSE	LagKE	EuARMSE	EuAFRMSE
GNOT	–	–	–	–
Transolver	–	–	–	–
Transolver++	–	–	–	–
LNO	5.032×10^{-4}	7.148×10^{-3}	2.779×10^{-4}	2.066×10^{-2}
AMG	–	–	–	–
Agent	$\mathbf{1.952 \times 10^{-4}}$	$\mathbf{2.813 \times 10^{-3}}$	$\mathbf{1.158 \times 10^{-4}}$	$\mathbf{7.134 \times 10^{-3}}$

Table 5: Transferability of **`LagInletOutlet`** to Lagrangian wave dynamics. We evaluate the agent-optimized architecture in an out-of-distribution setting on a novel 2D wave dataset. The agent-designed solver maintains superior accuracy across RMSE, LagKE, EuARMSE, and EuAFRMSE metrics compared to general-purpose baselines.

F SUPPLEMENTARY EXPERIMENTS ON LLM BACKBONE ANALYSIS

Tables 6, 7, and 8 detail supplementary results for our LLM backbone sensitivity analysis. Beyond RMSE, we incorporate additional metrics for the Eulerian and Lagrangian datasets that measure physical consistency and practical utility.

Model	RMSE	FRMSE	Kinetic Energy
gpt-5.5	5.066×10^{0}	4.218×10^{2}	3.514×10^{2}
gpt-4o	8.332×10^{0}	7.011×10^{2}	7.840×10^{2}
gpt-5.6-sol	5.537×10^{0}	4.670×10^{2}	4.035×10^{2}
gpt-5	5.771×10^{0}	5.147×10^{2}	3.308×10^{2}
claude-fable-5	6.766×10^{0}	5.525×10^{2}	5.142×10^{2}
claude-opus-4-8	6.687×10^{0}	5.473×10^{2}	5.676×10^{2}
grok-4.5	5.299×10^{0}	4.432×10^{2}	5.259×10^{2}
gemini-3.1-pro-preview	9.300×10^{0}	7.726×10^{2}	7.520×10^{2}
deepseek-v4-pro	3.685×10^{2}	9.608×10^{2}	2.015×10^{4}
qwen3.7-max	6.140×10^{0}	5.067×10^{2}	5.157×10^{2}
kimi-k2.7-code	5.784×10^{0}	4.783×10^{2}	4.215×10^{2}

Table 6: LLM backbone sensitivity on **`HighSpeedFlow`**. We evaluate alternative LLMs as the agent's backbone. The table reports RMSE alongside FRMSE and Kinetic Energy.

Model	RMSE	FRMSE	Kinetic Energy
gpt-5.5	1.345×10^{0}	1.078×10^{2}	2.784×10^{-1}
gpt-4o	4.056×10^{0}	3.518×10^{2}	2.507×10^{0}
gpt-5.6-sol	1.870×10^{0}	1.485×10^{2}	5.014×10^{-1}
gpt-5	4.790×10^{0}	2.801×10^{2}	1.990×10^{0}
claude-fable-5	3.435×10^{0}	2.691×10^{2}	1.693×10^{0}
claude-opus-4-8	2.200×10^{0}	1.710×10^{2}	6.128×10^{-1}
grok-4.5	2.002×10^{0}	1.594×10^{2}	5.295×10^{-1}
gemini-3.1-pro-preview	2.188×10^{0}	1.818×10^{2}	8.594×10^{-1}
deepseek-v4-pro	1.871×10^{0}	1.825×10^{2}	6.266×10^{-1}
qwen3.7-max	2.298×10^{0}	1.822×10^{2}	8.762×10^{-1}
kimi-k2.7-code	2.543×10^{0}	2.055×10^{2}	7.776×10^{-1}

Table 7: LLM backbone sensitivity on **`MixedParam`**. We evaluate alternative LLMs as the agent's backbone. The table reports RMSE alongside FRMSE and Kinetic Energy.

Model	RMSE	LagKE	EuARMSE	EuAFRMSE
gpt-5.5	5.530×10^{-1}	1.662×10^{3}	9.951×10^{-2}	2.299×10^{0}
gpt-4o	1.195×10^{1}	7.857×10^{5}	7.787×10^{0}	2.395×10^{2}
gpt-5.6-sol	5.647×10^{-1}	1.587×10^{3}	1.054×10^{-1}	2.467×10^{0}
gpt-5	1.194×10^{0}	1.304×10^{5}	6.773×10^{-1}	2.140×10^{1}
claude-fable-5	5.753×10^{-1}	2.149×10^{3}	1.073×10^{-1}	2.483×10^{0}
claude-opus-4-8	5.772×10^{-1}	2.225×10^{3}	1.103×10^{-1}	2.550×10^{0}
grok-4.5	5.808×10^{-1}	4.976×10^{3}	1.157×10^{-1}	2.768×10^{0}
gemini-3.1-pro-preview	5.848×10^{-1}	1.807×10^{3}	1.117×10^{-1}	2.606×10^{0}
deepseek-v4-pro	5.003×10^{-1}	1.252×10^{3}	7.332×10^{-2}	1.745×10^{0}
qwen3.7-max	1.195×10^{1}	7.856×10^{5}	7.787×10^{0}	2.395×10^{2}
kimi-k2.7-code	9.579×10^{-1}	1.064×10^{5}	5.447×10^{-1}	1.725×10^{1}

Table 8: LLM backbone sensitivity on **`LagInletOutlet`**. We evaluate alternative LLMs as the agent's backbone. The table reports RMSE alongside LagKE, EuARMSE, and EuAFRMSE to measure physical consistency and practical utility.

G Supplementary Experiment on Agent Long-Horizon Scalability

We present additional metrics for the agent long-horizon scalability experiments. The validation and test results at selected steps over extended iterations are detailed in Tables 9 and 10 for **HighSpeedFlow**, Tables 11 and 12 for **MixedParam**, and Tables 13 and 14 for **LagInletOutlet**. These tables include supplementary metrics for both the Eulerian and Lagrangian datasets. We observe that the trends in these additional metrics generally align with the primary metric, RMSE. This indicates that the agent's designs consistently improve over successive iterations not only in terms of point-wise error but also in physical consistency. These findings validate the promising potential of deploying LLM-driven autonomous research agents to address domain-specific challenges.

Iteration	RMSE	FRMSE	Kinetic Energy
1	4.654×10^0	3.976×10^2	3.373×10^2
2	4.301×10^0	3.647×10^2	2.987×10^2
3	4.194×10^0	3.518×10^2	2.802×10^2
4	4.135×10^0	3.460×10^2	2.717×10^2
5	4.112×10^0	3.421×10^2	2.678×10^2
10	4.189×10^0	3.455×10^2	2.657×10^2
15	4.077×10^0	3.405×10^2	2.582×10^2
20	3.903×10^0	3.224×10^2	2.406×10^2
25	3.977×10^0	3.298×10^2	2.421×10^2
30	3.960×10^0	3.279×10^2	2.235×10^2

Table 9: Long-horizon scalability on **HighSpeedFlow** (Validation). Supplementary metrics on the validation dataset for the agent evaluated over an extended horizon of 30 iterations. The agent directly optimizes its architecture based on validation performance and has no access to the test metrics.

Iteration	RMSE	FRMSE	Kinetic Energy
1	5.683×10^0	4.725×10^2	4.360×10^2
2	5.365×10^0	4.415×10^2	4.081×10^2
3	5.277×10^0	4.283×10^2	3.950×10^2
4	5.189×10^0	4.243×10^2	3.832×10^2
5	5.136×10^0	4.189×10^2	3.829×10^2
10	5.207×10^0	4.229×10^2	3.784×10^2
15	5.045×10^0	4.132×10^2	3.439×10^2
20	4.849×10^0	3.921×10^2	3.233×10^2
25	4.998×10^0	4.050×10^2	3.310×10^2
30	4.887×10^0	3.969×10^2	3.138×10^2

Table 10: Long-horizon scalability on **HighSpeedFlow** (Test). Supplementary metrics on the test dataset for the agent evaluated over an extended horizon of 30 iterations.

Iteration	RMSE	FRMSE	Kinetic Energy
1	9.274×10^{-1}	8.415×10^{1}	1.688×10^{-1}
2	9.387×10^{-1}	8.286×10^{1}	1.716×10^{-1}
3	9.327×10^{-1}	8.395×10^{1}	1.748×10^{-1}
4	1.284×10^{0}	1.188×10^{2}	3.057×10^{-1}
5	9.392×10^{-1}	8.450×10^{1}	1.780×10^{-1}
10	9.133×10^{-1}	8.192×10^{1}	1.673×10^{-1}
15	7.867×10^{-1}	7.750×10^{1}	1.466×10^{-1}
20	6.570×10^{-1}	6.759×10^{1}	9.796×10^{-2}
25	3.425×10^{-1}	3.087×10^{1}	5.273×10^{-2}
30	2.993×10^{-1}	2.685×10^{1}	5.242×10^{-2}

Table 11: Long-horizon scalability on **`MixedParam`** (Validation). Supplementary metrics on the validation dataset for the agent evaluated over an extended horizon of 30 iterations. During this process, the agent directly optimizes its architecture based on validation performance and has no access to the test metrics.

Iteration	RMSE	FRMSE	Kinetic Energy
1	9.043×10^{-1}	8.201×10^{1}	1.618×10^{-1}
2	9.171×10^{-1}	8.108×10^{1}	1.660×10^{-1}
3	9.056×10^{-1}	8.203×10^{1}	1.671×10^{-1}
4	1.258×10^{0}	1.175×10^{2}	2.976×10^{-1}
5	9.101×10^{-1}	8.215×10^{1}	1.705×10^{-1}
10	8.871×10^{-1}	7.983×10^{1}	1.608×10^{-1}
15	7.715×10^{-1}	7.638×10^{1}	1.405×10^{-1}
20	6.440×10^{-1}	6.663×10^{1}	9.264×10^{-2}
25	3.237×10^{-1}	2.922×10^{1}	5.032×10^{-2}
30	2.764×10^{-1}	2.476×10^{1}	5.018×10^{-2}

Table 12: Long-horizon scalability on **`MixedParam`** (Test). Supplementary metrics on the test dataset for the agent evaluated over an extended horizon of 30 iterations.

Iteration	RMSE	LagKE	EuARMSE	EuAFRMSE
1	5.802×10^{-1}	3.135×10^{3}	1.135×10^{-1}	2.698×10^{0}
2	5.781×10^{-1}	1.186×10^{3}	1.106×10^{-1}	2.582×10^{0}
3	5.779×10^{-1}	1.254×10^{3}	1.103×10^{-1}	2.574×10^{0}
4	5.781×10^{-1}	1.773×10^{3}	1.112×10^{-1}	2.611×10^{0}
5	5.780×10^{-1}	1.507×10^{3}	1.112×10^{-1}	2.609×10^{0}
10	5.815×10^{-1}	4.628×10^{3}	1.156×10^{-1}	2.765×10^{0}
15	6.144×10^{-1}	8.758×10^{3}	1.184×10^{-1}	2.835×10^{0}
20	6.119×10^{-1}	8.826×10^{3}	1.183×10^{-1}	2.846×10^{0}
25	5.851×10^{-1}	2.654×10^{3}	1.152×10^{-1}	2.705×10^{0}
30	6.127×10^{-1}	4.107×10^{3}	1.174×10^{-1}	2.788×10^{0}

Table 13: Long-horizon scalability on **`LagInletOutlet`** (Validation). Supplementary metrics on the validation dataset for the agent evaluated over an extended horizon of 30 iterations. During this process, the agent directly optimizes its architecture based on validation performance and has no access to the test metrics.

Iteration	RMSE	LagKE	EuARMSE	EuAFRMSE
1	5.776×10^{-1}	3.254×10^{3}	1.132×10^{-1}	2.698×10^{0}
2	5.755×10^{-1}	1.177×10^{3}	1.100×10^{-1}	2.572×10^{0}
3	5.753×10^{-1}	1.180×10^{3}	1.098×10^{-1}	2.562×10^{0}
4	5.756×10^{-1}	1.877×10^{3}	1.108×10^{-1}	2.606×10^{0}
5	5.756×10^{-1}	1.571×10^{3}	1.109×10^{-1}	2.605×10^{0}
10	5.788×10^{-1}	4.671×10^{3}	1.152×10^{-1}	2.760×10^{0}
15	6.124×10^{-1}	8.764×10^{3}	1.176×10^{-1}	2.812×10^{0}
20	6.099×10^{-1}	8.834×10^{3}	1.174×10^{-1}	2.817×10^{0}
25	5.825×10^{-1}	2.690×10^{3}	1.147×10^{-1}	2.694×10^{0}
30	6.107×10^{-1}	4.131×10^{3}	1.170×10^{-1}	2.779×10^{0}

Table 14: Long-horizon scalability on **`LagInletOutlet`** (Test). Supplementary metrics on the test dataset for the agent evaluated over an extended horizon of 30 iterations.

H Dataset Taxonomy

To characterize the challenges posed by AutoPDEBench, we organize the datasets along five categories:

- **Generalization.** Each dataset mixes problem instances that differ in PDE parameters, governing equations, boundary conditions, spatial dimension (1D, 2D, or 3D), resolution, or discretization. A single model must handle all of them.
- **Domains & Forcing.** The dynamics are shaped by complex domain and by external forcing.
- **Complex Physics.** The system consists of interacting components, develops discontinuities, or evolves randomly, properties that general-purpose solvers are not designed to capture.
- **Scale.** The problem is large enough, in the size of its state, the range of magnitudes it spans, or the length of its rollout, that computational cost and error accumulation become the primary obstacles.
- **Observational Irregularity.** The data offer only an imperfect view of the underlying system, with observations that are noisy, incomplete, irregularly sampled, or inconsistent across snapshots or between training and deployment, so the solver cannot assume complete, clean states on a fixed sampling pattern.

A dataset may fall into more than one category. Table 15 summarizes the assignment of each dataset. Datasets are grouped by category so that the table forms a block-diagonal structure, and datasets spanning two categories are placed at the boundary between the corresponding blocks.

Dataset	Generalization	Domains & Forcing	Complex Physics	Scale	Observational Irregularity
`MixedParam`	✓				
`MixedPDE`	✓				
`MixedDomain`	✓				
`MixedRes`	✓				
`MixedDim`	✓				
`MixedBC`	✓	✓			
`ComplexGeometry`		✓			
`ExtLag1`		✓			
`ExtLag2`		✓			
`LagFSI`		✓	✓		
`MultiPhase`			✓		
`Heterogeneous`			✓		
`MultiPhysics`			✓		
`HighSpeedFlow`			✓		
`Stochastic`			✓		
`Extreme`			✓	✓	
`HighDimensional`			✓	✓	
`3D`				✓	
`LargeLag`				✓	
`LongHorizon`				✓	
`RealData`					✓
`Sim2Real`					✓
`Partial`					✓
`IrregularTime`					✓
`LagInletOutlet`					✓

Table 15: Taxonomy of AutoPDEBench datasets.

I DATASETS

In the sections below, we provide a comprehensive breakdown of each dataset introduced in our study. To keep the manuscript's file size manageable, all embedded figures have been intentionally compressed; however, readers can access the high-resolution, vectorized versions directly through our associated GitHub repository. Because our research spans a broad spectrum of distinct physical systems, maintaining a single, globally unified mathematical glossary across the entire paper was impractical. Therefore, mathematical symbols are strictly self-contained within their respective sections, meaning a variable used to denote one physical property in a given scenario may be redefined to represent an entirely different quantity in the next.

The datasets curated for this study are either adapted directly from peer-reviewed literature or numerically simulated utilizing well-established numerical schemes. To guarantee the highest degree of fidelity and reliability in our simulated datasets, we implemented a rigorous computational validation protocol. All underlying numerical solvers were systematically benchmarked against known analytical solutions for canonical problems and subjected to comprehensive spatial and temporal convergence studies to verify that our results are grid-independent.

I.1 HIGHSPEEDFLOW

Dataset Description: In this 2D circular blast model, a central high-pressure zone creates an outward shock wave and an inward expansion wave. The shock wave repeatedly reflects back and forth between the boundaries and the origin, losing strength with each bounce. As these reflecting shocks interact with the post-shock gas, they generate turbulent instabilities.

Modeling Objective: Let $x_{t_1} \in \mathbb{R}^{4\times128\times128}$ denote the state of the system at time t_1. The three feature channels represent the physical fields of temperature, density, and the x and y velocity fields, respectively, evaluated across a 128×128 spatial discretization grid. The initial time $t_1 \in \mathbb{R}^1$ and the target time $t_2 \in \mathbb{R}^1$ are explicitly provided as inputs to the model. Our objective is to develop a neural simulator, defined as a parameterized function Ψ, that maps the initial system state and the given time coordinates to the subsequent state at t_2. Formally, we aim to predict $\hat{x}_{t_2} = \Psi(x_{t_1}, t_1, t_2)$, where the predicted output $\hat{x}_{t_2}$ retains the same spatial and feature dimensions as the input state.

Dataset Specific Challenges: Existing neural solvers are designed for low-Mach-number regimes, where fluid dynamics generally exhibit smooth temporal evolution. In contrast, compressible flows approaching or exceeding the speed of sound develop abrupt, highly non-linear discontinuities, such as shock waves. Additionally, because the underlying numerical solver must adapt its resolution to capture these sudden physical changes, the time step ($\Delta t = t_2 - t_1$) between the input and output states in this dataset is inherently non-uniform.

Dataset Generation: We adopt the Circular Blast dataset from Helwig et al. (2026). The circular blast dataset is generated using the HyBurn code, a finite volume solver that uses low-dissipation, WENO-based high-order Godunov methods with adaptive mesh refinement via the AMReX library (Zhang et al., 2019). HyBurn implements high-order Godunov schemes and uses an explicit third-order three-stage Runge-Kutta time-stepping algorithm along with two independent CFL numbers: one for the hyperbolic and the other for the parabolic terms of the compressible Navier-Stokes equations, with the actual timestep being whichever is smaller. For the circular blast specifically, the authors initialize a circular high-pressure region over a quarter domain with symmetry boundaries, varying the initial pressure ratio from 1.99 to 50 across 99 cases. They use two Adaptive Mesh Refinement (AMR) levels triggered by predefined pressure and density ratio thresholds between adjacent cells.

Governing Physics: The circular blast case is governed by the two-dimensional compressible Navier–Stokes equations for a single gaseous phase, written in the conservative form used by finite-volume Godunov solvers as

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} + \frac{\partial \mathbf{G}}{\partial y} = \frac{\partial \mathbf{F}_v}{\partial x} + \frac{\partial \mathbf{G}_v}{\partial y}, \tag{19}$$

where $\mathbf{U}$ is the vector of conserved variables, $\mathbf{F}$ and $\mathbf{G}$ are the inviscid (hyperbolic) fluxes, and $\mathbf{F}_v$ and $\mathbf{G}_v$ are the viscous (parabolic) fluxes; this hyperbolic–parabolic split is precisely why the solver carries two independent CFL numbers. The conserved state is $\mathbf{U} = (\rho,\ \rho u,\ \rho v,\ \rho E)^\top$, with density

ρ, velocity components u and v, and total energy per unit mass $E = e + \frac{1}{2}(u^2 + v^2)$ for specific internal energy e. The inviscid fluxes are

$$\mathbf{F} = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ (\rho E + p)\, u \end{pmatrix}, \qquad \mathbf{G} = \begin{pmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ (\rho E + p)\, v \end{pmatrix}, \tag{20}$$

and the viscous fluxes are

$$\mathbf{F}_v = \begin{pmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ u\,\tau_{xx} + v\,\tau_{xy} - q_x \end{pmatrix}, \qquad \mathbf{G}_v = \begin{pmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ u\,\tau_{xy} + v\,\tau_{yy} - q_y \end{pmatrix}. \tag{21}$$

The system is closed with a Newtonian viscous stress tensor under Stokes' hypothesis,

$$\tau_{ij} = \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - \frac{2}{3} \mu\, (\nabla \cdot \mathbf{u})\, \delta_{ij}, \tag{22}$$

with heat flux given by Fourier's law $q_x = -k\, \partial T / \partial x$ and $q_y = -k\, \partial T / \partial y$, where μ is the dynamic viscosity and k the thermal conductivity. Finally, since the gas is modeled as ideal, the equation of state, internal energy, and local sound speed are

$$p = \rho R T, \qquad e = \frac{p}{(\gamma - 1)\rho}, \qquad a = \sqrt{\gamma R T}, \tag{23}$$

where R is the specific gas constant and γ the ratio of specific heats; this sound speed a feeds the maximum wave speed $\lambda = \max(|u| + a,\ |v| + a)$ that governs the CFL timestep condition.

Experiment Results The complete experimental results are presented in Table 16. The dataset is represented on an Eulerian grid, and we employ RMSE, FRMSE, and Kinetic Energy as our evaluation metrics. The results demonstrate that the proposed automated research agent significantly outperforms general-purpose baseline solvers.

	RMSE	FRMSE	Kinetic Energy
GNOT	1.826×10^1	1.529×10^3	1.538×10^3
Transolver	3.232×10^1	2.372×10^3	3.067×10^3
Transolver++	3.271×10^1	2.406×10^3	3.373×10^3
LNO	4.482×10^1	3.282×10^3	6.092×10^3
AMG	2.819×10^1	2.181×10^3	3.255×10^3
Agent	$\mathbf{5.066 \times 10^0}$	$\mathbf{4.218 \times 10^2}$	$\mathbf{3.514 \times 10^2}$

Table 16: Performance comparison on the **`HighSpeedFlow`** dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 7 for visualizations of the dataset.

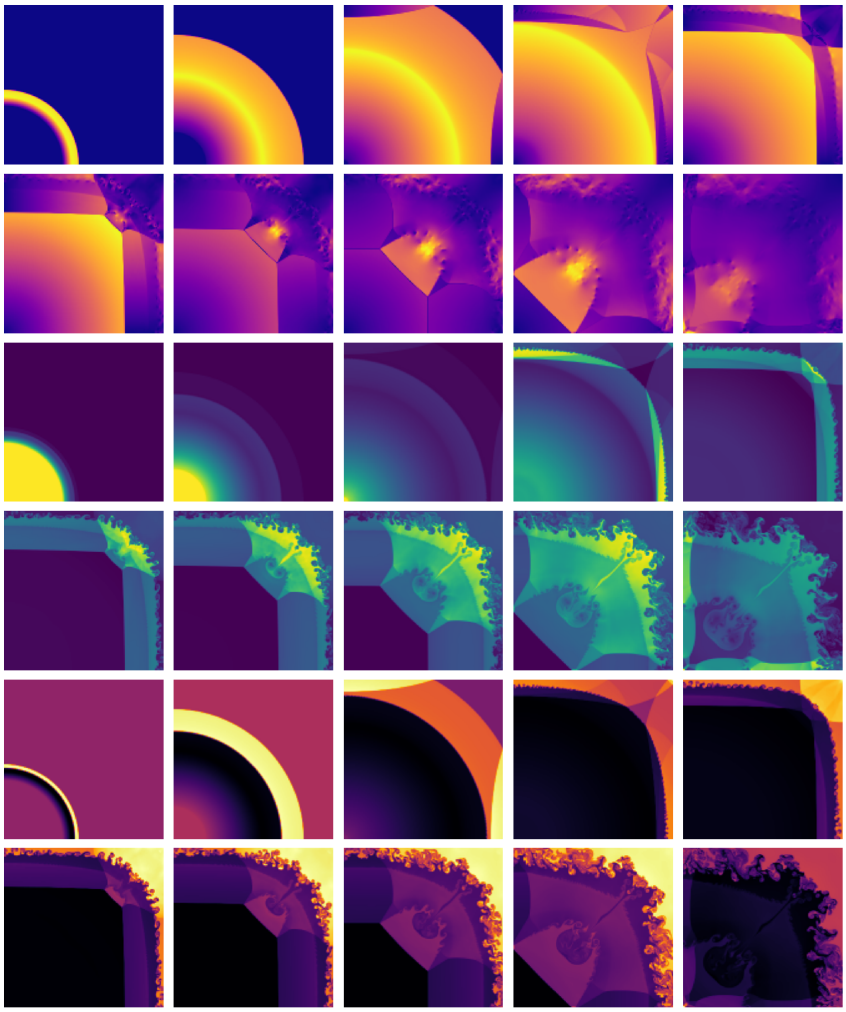


Figure 7: Visualization of the **HighSpeedFlow** dataset. The first two rows display velocity norm, the middle two rows show density, and the bottom two rows show temperature.

I.2 REALDATA

Dataset Description: This dataset captures the fluid dynamics of laminar to turbulent flow around a cylinder. While the simulation resolves the full three-dimensional flow field, we extracted only a 2D cross section for this dataset. A cylinder with a diameter 30 mm is positioned centrally within the domain, serving as a bluff body that obstructs the incoming fluid stream. As the fluid encounters this obstacle, it creates a distinct pressure gradient and a downstream wake region characterized by periodic vortex shedding.

Modeling Objective: The state of the system is represented by the tensor $x_t \in \mathbb{R}^{2\times128\times128}$. The two feature channels correspond specifically to the horizontal and vertical axis velocity fields, evaluated across a 128×128 spatial discretization grid. The objective is to develop a neural simulator, defined as a parameterized function Ψ, that maps the current system state to the state at the subsequent timestep. Formally, we aim to predict the next state $\hat{x}_{t+1} = \Psi(x_t)$.

Dataset Specific Challenges: The dataset's primary challenge is the inherent measurement noise typical of real-world sensors. Unlike idealized synthetic data, these fluctuations can obscure the underlying physical laws, making it difficult for models to distinguish between meaningful fluid dynamics and measurement error. Existing neural models often struggle with this data because they are typically designed for clean, high-fidelity simulations. Consequently, they may overfit to the noise or fail to maintain temporal stability.

Dataset Generation: We adopt the dataset from Hu et al. (2026). The real-world Cylinder data is measured in a circulating water tunnel using Particle Image Velocimetry (PIV). The water tunnel generates an incoming flow that interacts with the cylinder, and the water is seeded with fluorescent particles that are illuminated by a continuous laser forming a thick sheet on the imaging plane, so that the fluid velocity can be inferred from the motion of these particles. The particle motions are recorded with high-speed cameras, and the resulting images are then processed with PIVLab to recover the flow-velocity field at each time step.

Governing Physics: In the idealized continuum limit, the wake behind the circular cylinder is a low-speed, effectively incompressible water flow and should therefore be described by the incompressible Navier–Stokes equations,

$$\nabla \cdot \mathbf{u} = 0, \tag{24}$$

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\,\mathbf{u} = -\frac{1}{\rho}\nabla p + \nu\,\nabla^2 \mathbf{u}, \tag{25}$$

where $\mathbf{u}(\mathbf{x}, t)$ is the velocity field, p the pressure, ρ the (constant) fluid density, and ν the kinematic viscosity. The first equation enforces incompressibility and the second expresses momentum conservation, with the dynamics governed by the Reynolds number $\mathrm{Re} = UD/\nu$ formed from the free-stream speed U and the cylinder diameter D; as Re increases, the wake transitions from steady flow, through the periodic Kármán vortex street, toward turbulence.

The measured data, however, are not exact solutions of these equations. Particle Image Velocimetry does not observe $\mathbf{u}$ directly but instead estimates it from the displacement of tracer particles over a finite time interval, and it recovers only the in-plane velocity while the pressure p remains unobserved. It is therefore natural to model the measured field as

$$\mathbf{u}_{\text{meas}}(\mathbf{x}, t) = \mathbf{u}(\mathbf{x}, t) + \boldsymbol{\varepsilon}(\mathbf{x}, t), \tag{26}$$

where $\mathbf{u}$ satisfies the equations above and $\boldsymbol{\varepsilon}$ is a random measurement error. This error is approximately zero-mean but fluctuates independently from frame to frame and across interrogation windows, and it arises from several stochastic sources: sensor (camera) noise in the recorded particle images, the finite number of tracer particles sampled within each interrogation window, the loss of particles that drift out of the finite-thickness laser sheet between exposures, and sub-pixel error in locating the cross-correlation peak. In addition, the incoming flow cannot be held perfectly uniform, so the effective inflow boundary condition itself carries a small stochastic component. Because of the random measurement noise, together with the finite spatial and temporal sampling and the spatial averaging inherent to the interrogation windows, the data satisfy the Navier–Stokes equations only approximately, which is precisely the gap between real-world and simulated data that the benchmark is designed to expose.

Experiment Results: Table 17 details the comprehensive experimental results. Under an Eulerian grid formulation, we assess performance based on RMSE, FRMSE, and Kinetic Energy. These metrics indicate that the custom design of the automated research agent yields substantial improvements over standard, general-purpose solvers.

	RMSE	FRMSE	Kinetic Energy
GNOT	5.638×10^{-3}	3.059×10^{-1}	5.609×10^{-4}
Transolver	5.626×10^{-3}	2.916×10^{-1}	5.516×10^{-4}
Transolver++	5.618×10^{-3}	2.906×10^{-1}	5.582×10^{-4}
LNO	6.946×10^{-3}	4.411×10^{-1}	6.585×10^{-4}
AMG	4.889×10^{-3}	2.944×10^{-1}	5.330×10^{-4}
Agent	$\mathbf{4.458 \times 10^{-3}}$	$\mathbf{2.861 \times 10^{-1}}$	$\mathbf{4.430 \times 10^{-4}}$

Table 17: Performance comparison on the `RealData` dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 8 for visualizations of the dataset. We provide examples on the 2D cross-sectional velocity measurements.

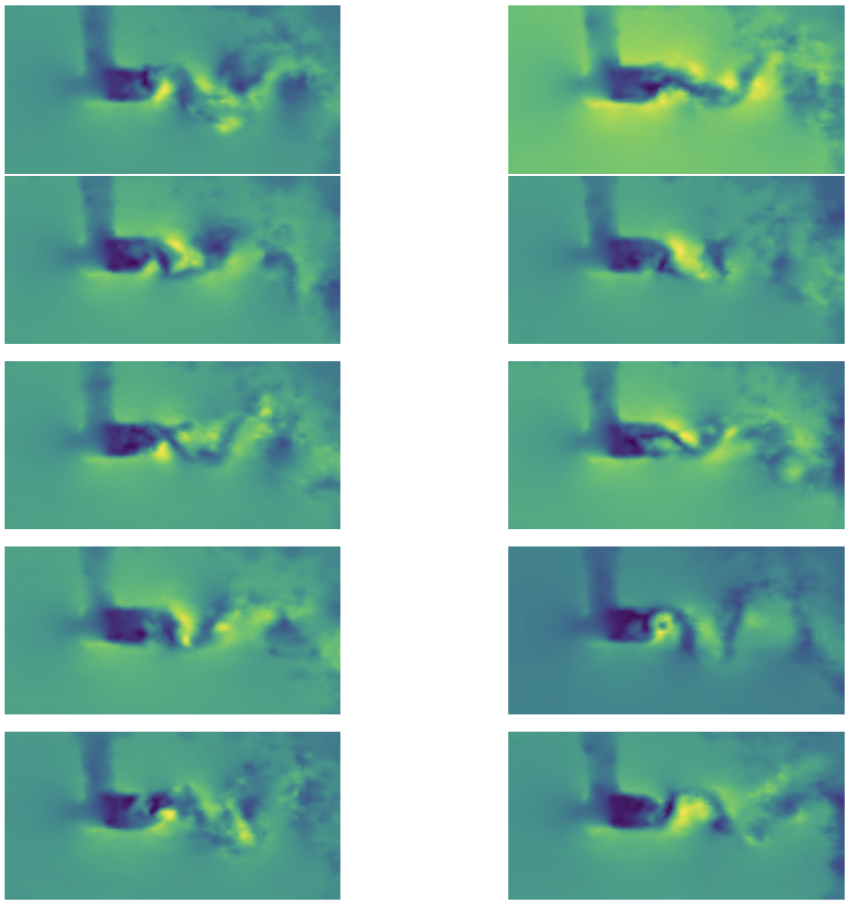


Figure 8: Visualization of the `RealData` dataset. This dataset provides 2D cross-sectional velocity measurements of the flow around a cylinder.

I.3 MIXEDPARAM

Dataset Description: This dataset captures the parameter-dependent fluid dynamics governed by the incompressible Navier-Stokes equations. The fluid behavior spans a wide spectrum of physical regimes, transitioning from steady, laminar flows to highly unsteady, chaotic turbulence depending on the varying Reynolds number (Re). The training data encompasses a discrete set of simulations with Reynolds numbers sampled at intervals of 100, defined as $Re \in \{100, 200, \ldots, 5000\}$. To rigorously evaluate model generalization, the validation and test sets feature a finer sampling interval of 25, defined as $Re \in \{100, 125, 150, \ldots, 5000\}$.

Modeling Objective: The state of the fluid system is represented by the tensor $x_t \in \mathbb{R}^{1\times128\times128}$, where the single feature channel corresponds to the vorticity field evaluated across a 128×128 spatial discretization grid. The system is explicitly conditioned on the PDE parameter tensor $p \in \mathbb{R}^1$, representing the Reynolds number of the specific simulation. The objective is to develop a parameter-conditioned neural simulator, defined as a function Ψ, that maps the current system state and the governing parameter to the state at the subsequent timestep. Formally, we aim to predict the next state $\hat{x}_{t+1} = \Psi(x_t, p)$.

Dataset Specific Challenges: The dataset's primary challenge lies in the architectural design required to effectively embed the PDE parameter into the neural model. The model must learn to dynamically adapt its localized spatial filters based on a global scalar value. Existing neural operators often struggle with efficiently fusing this lower-dimensional parameter into the high-dimensional spatial feature space. Furthermore, because the test set contains intermediate Reynolds numbers unseen during training, the model must achieve robust zero-shot generalization to out-of-distribution parameters.

Dataset Generation: The dataset is generated by directly simulating the two-dimensional incompressible Navier–Stokes equations in vorticity–streamfunction form using the pseudo-spectral method. The stream function and velocity are recovered from the vorticity in Fourier space, the nonlinear advection term is evaluated in physical space and transformed back, and aliasing is controlled with the 2/3 rule. Time integration uses a semi-implicit scheme that treats the viscous term with Crank–Nicolson and the nonlinear advection plus forcing with Heun's method (a two-stage, second-order predictor–corrector), with an adaptive step size set by a CFL condition combining an advective limit and a diffusive limit and taking the smallest step across the batch.

Governing Physics: The dataset is governed by the two-dimensional incompressible Navier–Stokes equations on the doubly-periodic domain $\Omega = [0, 2\pi]^2$. In velocity–pressure form these read

$$\nabla \cdot \mathbf{u} = 0, \tag{27}$$

$$\partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla p + \frac{1}{\mathrm{Re}} \nabla^2 \mathbf{u} + \mathbf{F}, \tag{28}$$

where $\mathbf{u}(\mathbf{x}, t)$ is the velocity field, p the pressure, $\mathbf{F}$ an external body force, and Re the Reynolds number. Taking the scalar curl of the momentum equation eliminates the pressure and, using incompressibility, yields the equivalent vorticity–stream-function formulation that the solver actually integrates, in which the vorticity $\omega = \nabla \times \mathbf{u} = \partial_x u_2 - \partial_y u_1$ evolves as

$$\partial_t \omega + (\mathbf{u} \cdot \nabla)\omega = \frac{1}{\mathrm{Re}} \nabla^2 \omega + f, \tag{29}$$

where $f = \nabla \times \mathbf{F}$ is the corresponding vorticity forcing. The velocity is recovered at each instant from a stream function ψ through

$$\mathbf{u} = \nabla^{\perp}\psi = (\partial_y \psi,\ -\partial_x \psi), \tag{30}$$

$$\omega = -\nabla^2 \psi, \tag{31}$$

which enforces $\nabla \cdot \mathbf{u} = 0$ automatically. The flow is driven by the steady Kolmogorov forcing

$$f(x, y) = -4\cos(4y), \tag{32}$$

a unidirectional shear forcing that is sinusoidal in the transverse coordinate y with wavenumber 4, so that increasing Re drives the resulting flow from a smooth laminar state toward two-dimensional

turbulence. Together with periodic boundary conditions on Ω and a prescribed initial vorticity field $\omega(\cdot, 0)$, these equations fully determine the dynamics recorded in the dataset.

Experiment Results: Complete experimental results are detailed in Table 18. Performance on the Eulerian grid dataset was assessed using RMSE, FRMSE, and Kinetic Energy metrics, revealing that the automated research agent yields significantly higher accuracy than standard baseline solvers.

	RMSE	FRMSE	Kinetic Energy
GNOT	3.417×10^0	2.744×10^2	1.845×10^0
Transolver	4.359×10^0	3.514×10^2	3.593×10^0
Transolver++	4.329×10^0	3.496×10^2	3.576×10^0
LNO	4.346×10^0	3.553×10^2	3.605×10^0
AMG	3.128×10^0	2.688×10^2	1.741×10^0
Agent	$\mathbf{1.345 \times 10^0}$	$\mathbf{1.078 \times 10^2}$	$\mathbf{2.780 \times 10^{-1}}$

Table 18: Performance comparison on the **MixedParam** dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 9 for visualizations of the dataset. Each row represents a trajectory with unique Reynold number.

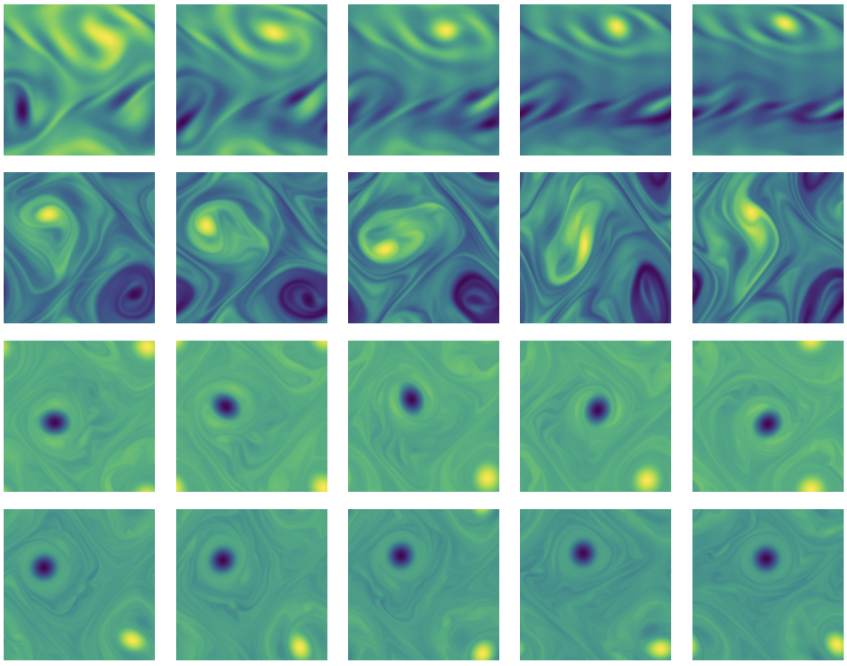


Figure 9: Visualization of the **MixedParam** dataset. This is the Navier-Stokes Equation with mixed Reynold number. The four rows correspond to Reynold number 100, 1000, 2500, and 5000, respectively.

I.4 HETEROGENEOUS

Dataset Description: This dataset comprises simulations of the Rayleigh–Taylor instability, a fundamental hydrodynamic phenomenon that arises when a denser fluid is accelerated into a lighter one. Under gravity or an equivalent acceleration, small perturbations along the fluid interface grow over time, producing the characteristic morphology of the instability: rising bubbles of the lighter fluid interpenetrating with sinking spikes of the heavier fluid. A defining feature of this dataset is that the fluid densities vary across trajectories, which induces a broad spectrum of dynamical regimes. Trajectories with heavier top fluids exhibit faster sinking and develop increasingly chaotic, finely structured patterns, whereas those with lighter top fluids evolve more slowly and yield smoother, more coherent flows.

Modeling Objective: The goal is to learn a neural simulator Ψ that advances the particle system by one timestep. At time t, the input is a pair of tensors $\{V, A\}$ derived from the recent history of N particles connected by M edges. Starting from the raw positional history $p_{t-5:t} \in \mathbb{R}^{6\times N\times 2}$ and the time-invariant particle attributes $A \in \mathbb{R}^{N\times 1}$, we assemble the node feature matrix $V \in \mathbb{R}^{N\times 13}$ by concatenating the current position p_t, the five most recent velocity vectors $v_{t-4:t}$, and the particle attribute. The simulator then maps these inputs to a prediction of the position at the subsequent timestep,

$$\hat{p}_{t+1} = \Psi(V, A), \tag{33}$$

where $\hat{p}_{t+1} \in \mathbb{R}^{N\times 2}$ denotes the predicted per-particle position. Rolling this one-step predictor forward autoregressively then reconstructs the full trajectory.

Dataset Specific Challenges: The principal difficulty posed by this dataset stems from its heterogeneous physical properties, which govern markedly non-uniform dynamical behaviors that standard neural simulators struggle to resolve. Because the density contrast varies from trajectory to trajectory, a single model must internalize an entire family of attribute-conditional physical laws rather than a fixed dynamics. A conventional approach that merely concatenates these attributes into the node feature vector frequently proves insufficient: the attribute is treated as one weakly weighted input among many, and the model fails to learn how the governing dynamics should change as a function of it. Effectively conditioning the learned dynamics on the underlying physical attributes is therefore essential to capturing the full range of behaviors, from the smooth evolution of low-contrast flows to the chaotic mixing of high-contrast ones.

Dataset Generation: The dataset is generated by simulating the two-phase Rayleigh–Taylor instability with Smoothed Particle Hydrodynamics (SPH) (Gingold & Monaghan, 1977) using the PySPH framework (Ramachandran et al., 2021), in which the fluid is discretized into Lagrangian particles rather than solved on a fixed grid. The two fluids occupy a rectangular domain of width $L_x = 1$ and height $L_y = 2$ under a downward gravitational acceleration $g_y = -1$, with a heavier phase of density ρ_1 initially resting on a lighter phase of density ρ_2. The interface is perturbed by a single sinusoidal mode, with particles assigned to the upper phase wherever $y > 1 - 0.15\sin(2\pi x)$, so that the instability grows from this controlled initial disturbance; the domain is enclosed by solid boundary particles built from a ghost region five particle-spacings thick. The particles are laid out on a uniform lattice at resolution $n_x = 50$ (spacing $\Delta x = L_x/n_x$) with smoothing length $h = 1.2\,\Delta x$ and per-particle mass set to reproduce each phase's reference density. The dynamics are advanced with the Transport-Velocity Formulation (TVF) scheme, a weakly-compressible SPH method that transports particles with a modified (transport) velocity to keep them evenly distributed and suppress the tensile instability; incompressibility is approximated through an artificial equation of state $p = c_0^2\rho$ with an artificial sound speed $c_0 = V_{\max}/\mathrm{Fr}$ chosen from a low Mach/Froude number $\mathrm{Fr} = 0.01$ (using $V_{\max} = \sqrt{0.5\,L_y|g_y|}$), and the kinematic viscosity $\nu = V_{\max}L_y/\mathrm{Re}$ is fixed by a Reynolds number $\mathrm{Re} = 420$. Time integration uses an adaptive step taken as half the minimum of a CFL (acoustic) limit $0.25\,h/(c_0 + V_{\max})$, a viscous limit $0.125\,h^2/\nu$, and a body-force limit $0.25\sqrt{h/|g_y|}$, integrating to a final time $t_f = 25$ and writing particle snapshots periodically.

Governing Physics: The dataset models the Rayleigh–Taylor instability of two immiscible, incompressible, viscous fluids in a gravitational field, governed by the variable-density incompressible Navier–Stokes equations on the domain $\Omega = [0, L_x] \times [0, L_y]$ under the body force $\mathbf{g} = (0, g_y)$,

$$\nabla\cdot\mathbf{u} = 0, \qquad \frac{D\rho}{Dt} = 0, \qquad \rho\,\frac{D\mathbf{u}}{Dt} = -\nabla p + \nabla\cdot\big(2\mu\,\mathbf{S}\big) + \rho\,\mathbf{g}, \tag{34}$$

where $\frac{D}{Dt} = \partial_t + \mathbf{u} \cdot \nabla$ is the material derivative, $\mathbf{u}$ the velocity, p the pressure, ρ the density, $\mathbf{S} = \frac{1}{2}\big(\nabla\mathbf{u} + \nabla\mathbf{u}^\top\big)$ the strain-rate tensor, and $\mu = \rho\nu$ the dynamic viscosity for a common kinematic viscosity ν. The density is piecewise constant, taking the value ρ_1 in the upper (heavy) fluid and ρ_2 in the lower (light) fluid with $\rho_1 > \rho_2$, and the two phases meet across a material interface $\Gamma(t)$ advected by the flow. The system starts from rest, $\mathbf{u}(\cdot, 0) = \mathbf{0}$, with the interface given the single-mode sinusoidal perturbation

$$\Gamma(0) = \{\, y = \eta(x) \,\}, \qquad \eta(x) = 1 - 0.15\, \sin(2\pi x), \tag{35}$$

so that $\rho(\mathbf{x}, 0) = \rho_1$ for $y > \eta(x)$ and ρ_2 for $y < \eta(x)$. Across $\Gamma(t)$ the velocity and the traction are continuous (there is no surface tension),

$$[\![\, \mathbf{u} \,]\!] = \mathbf{0}, \qquad \big[\!\big[\, (-p\,\mathbf{I} + 2\mu\,\mathbf{S})\,\mathbf{n} \,\big]\!\big] = \mathbf{0}, \tag{36}$$

and no-slip conditions $\mathbf{u} = \mathbf{0}$ are imposed on the solid walls $\partial\Omega$. The configuration is gravitationally unstable—a denser fluid supported above a lighter one—so the perturbation is amplified at a rate set by the Atwood number, while the flow regime is fixed by the Reynolds and Froude numbers,

$$A = \frac{\rho_1 - \rho_2}{\rho_1 + \rho_2}, \qquad \mathrm{Re} = \frac{V_{\max} L_y}{\nu}, \qquad \mathrm{Fr} = \frac{V_{\max}}{c_0}, \qquad V_{\max} = \sqrt{\tfrac{1}{2} L_y\, |g_y|}. \tag{37}$$

In the weakly-compressible formulation actually integrated, the strict constraint $\nabla \cdot \mathbf{u} = 0$ is relaxed to the continuity equation $\frac{D\rho}{Dt} = -\rho\, \nabla \cdot \mathbf{u}$ closed by the stiff artificial equation of state $p = c_0^2\, \rho$, in which the artificial sound speed $c_0 = V_{\max}/\mathrm{Fr}$ is chosen large enough ($\mathrm{Fr} = 0.01 \ll 1$) that density fluctuations remain of order Fr^2 and the incompressible limit is recovered.

Experiment Results: We present the complete results in Table 19. Because the datasets consist of Lagrangian particles, we evaluate performance using the RMSE of particle positions, Lagrangian Kinetic Energy (LagKE), and Eulerian aggregated RMSE and FRMSE (EuARMSE, EuAFRMSE). The results demonstrate that the automated research agent yields significantly higher accuracy than standard baseline solvers.

	RMSE	LagKE	EuARMSE	EuAFRMSE
GNOT	4.207×10^{-3}	1.324×10^{-1}	6.836×10^{-4}	2.170×10^{-2}
Transolver	4.155×10^{-3}	1.374×10^{-1}	6.655×10^{-4}	2.169×10^{-2}
Transolver++	3.963×10^{-3}	1.622×10^{-1}	6.280×10^{-4}	2.186×10^{-2}
LNO	3.951×10^{-3}	1.776×10^{-1}	7.030×10^{-4}	2.590×10^{-2}
AMG	3.900×10^{-3}	1.683×10^{-1}	6.052×10^{-4}	2.155×10^{-2}
Agent	$\mathbf{3.311 \times 10^{-3}}$	$\mathbf{8.797 \times 10^{-2}}$	$\mathbf{5.280 \times 10^{-4}}$	$\mathbf{1.585 \times 10^{-2}}$

Table 19: Performance comparison on the **Heterogeneous** dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 10 for visualizations of the dataset.

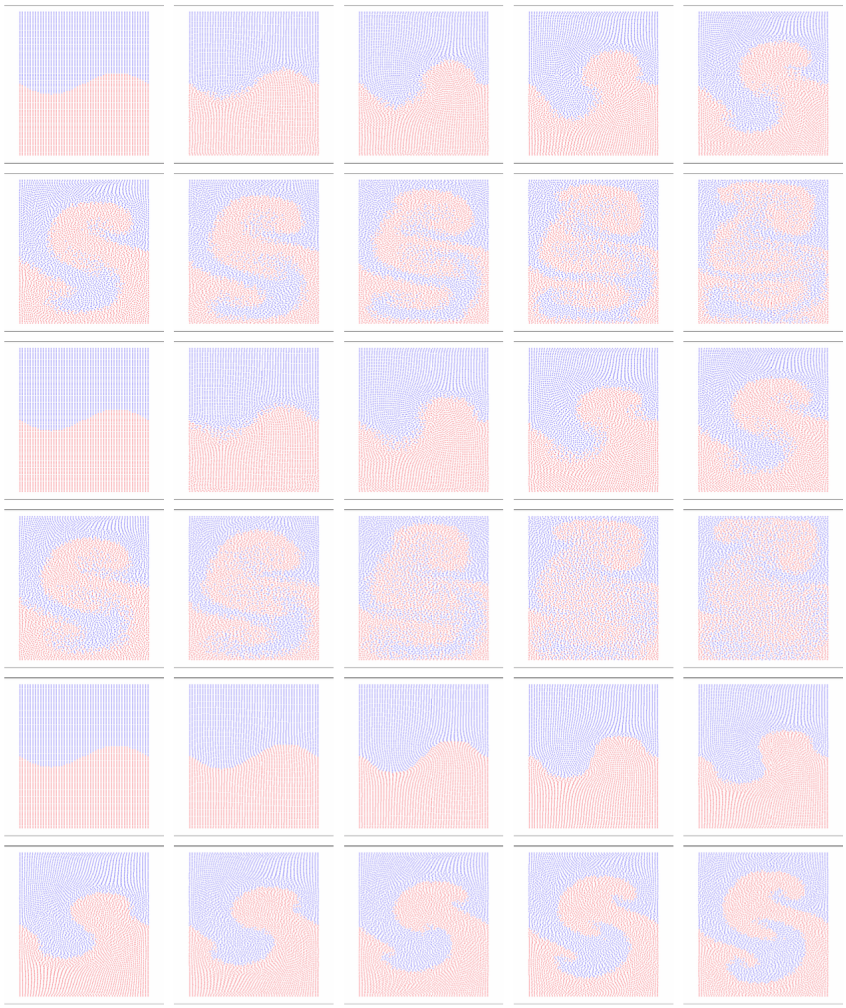


Figure 10: Visualization of the **Heterogeneous** dataset. This is the Reyleigh Taylor Instability in Lagrangian particle representation. The density of particles differ across trajectories, resulting in distinct behaviors.

I.5 PARTIAL

Dataset Description: This dataset captures the fluid dynamics of a two-dimensional incompressible flow governed by the Navier-Stokes equations, at Reynolds number of $Re = 1000$. This dataset models a sensor-limited scenario where the physical domain is only partially observable.

Modeling Objective: The underlying full physical state is represented by the tensor $x_t \in \mathbb{R}^{1\times128\times128}$, denoting the complete vorticity field. The model also receives a binary mask tensor $m \in \{0,1\}^{1\times128\times128}$, where a value of 1 indicates an available measurement and 0 indicates missing data. The objective is to develop a neural simulator, Ψ, that maps the incomplete current observation to the complete, unmasked state at the subsequent timestep. Formally, we aim to predict the full future field $\hat{x}_{t+1} = \Psi(X_t \odot m)$.

Dataset Specific Challenges: The primary challenge is that partial observability transforms a deterministic physical system into a fundamentally ill-posed and non-deterministic learning problem. Because critical spatial information is hidden, multiple physically valid full-state configurations could theoretically correspond to the same masked observation. Consequently, the neural model must simultaneously solve an ambiguous spatial inverse problem to infer the unobserved regions while computing the temporal evolution of the system.

Dataset Generation: In the first stage, a two-dimensional incompressible Navier–Stokes flow in vorticity–stream-function form, driven by the steady Kolmogorov forcing $f(x, y) = -4\cos(4y)$ on the doubly-periodic domain $[0, 2\pi]^2$, is advanced with a pseudo-spectral solver at a fixed Reynolds number $\mathrm{Re} = 1000$; each of the 50 trajectories is initialized from a Gaussian random field and integrated for 100 time units so the flow reaches a statistically developed state before recording, after which the vorticity is simulated at 256×256 resolution and stored, spatially subsampled by a factor of two, on a 128×128 grid with snapshots saved every $\Delta t = 0.1$ up to $T = 20$. In the second stage the dataset is made partially observed: a binary mask $m \in \{0,1\}^{1\times128\times128}$, with $m = 1$ where a measurement is available and $m = 0$ where data are missing, is generated at random during post-processing and applied through the element-wise product, so that only the retained entries of the field are visible while the true underlying state is preserved as ground truth.

Governing Physics: The governing physics is the two-dimensional incompressible Navier–Stokes equation in vorticity–stream-function formulation similar to the one described in Section I.3

Experiment Results: As detailed in Table 20, the complete experimental results on the Eulerian grid demonstrate the superiority of the automated research agent's architecture. Measured by RMSE, FRMSE, and Kinetic Energy, the specialized design surpasses general-purpose baselines.

	RMSE	FRMSE	Kinetic Energy
GNOT	1.978×10^{0}	1.875×10^{2}	9.480×10^{-1}
Transolver	3.636×10^{0}	4.224×10^{2}	4.509×10^{0}
Transolver++	3.636×10^{0}	4.224×10^{2}	4.519×10^{0}
LNO	3.636×10^{0}	4.223×10^{2}	4.505×10^{0}
AMG	2.111×10^{0}	2.031×10^{2}	9.980×10^{-1}
Agent	$\mathbf{8.633 \times 10^{-2}}$	$\mathbf{7.683 \times 10^{0}}$	$\mathbf{2.100 \times 10^{-2}}$

Table 20: Performance comparison on the **Partial** dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 11 for visualizations of the dataset.

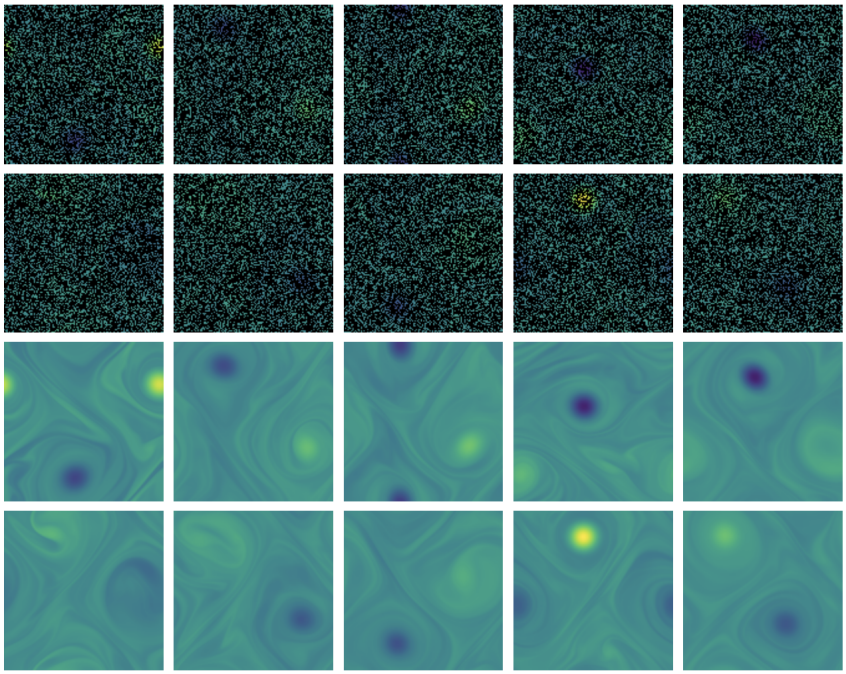


Figure 11: Visualization of the **Partial** dataset.The first two rows show partially observed data, while the bottom two show complete data.

I.6 MIXEDDOMAIN

Dataset Description: This dataset provides a multi-representation view of fluid flow around a cylinder. It captures the same underlying physics through three distinct lenses: a structured Eulerian grid, an unstructured Eulerian mesh, and a Lagrangian particle-based representation. The grid data consists of a 64×32 uniformly spaced domain, while the mesh data utilizes 2048 randomly scattered points to sample the flow field. For the Lagrangian perspective, the dataset tracks 16,000 individual particles as they advect through the domain. Because all three describe identical dynamics and differ only in how the flow field is discretized, the aim is to develop a generalized neural solver that can ingest any one of these representations and produce predictions in that same representation.

Modeling Objective: The objective is to develop a unified neural simulator, Ψ, capable of cross-domain forecasting through a single, representation-agnostic input format. Let $x_t \in \mathbb{R}^{N\times 13}$ denote the system state at time t, where the node count N varies by representation (grid points, mesh points, or Lagrangian particles). The 13 input features per node comprise the position (2), a five-step velocity history ($2 \times 5 = 10$), and a scalar identifier indicating which representation the node belongs to (1). This scalar is encoded as 0 for the irregular mesh, 1 for the regular grid, and 2 or 3 for Lagrangian particles, where the latter two values further distinguish fluid from solid particles. For Lagrangian particles the position denotes the particle location at the current step, whereas for the grid and mesh it is the fixed node position. The goal is to learn the parameterized solver Ψ such that

$$\hat{y}_{t+1} = \Psi(x_t), \tag{38}$$

where $\hat{y}_{t+1} \in \mathbb{R}^{N\times 2}$ is the predicted next-step velocity for every node. The model can therefore take any representation as input and forecast the velocity field in whichever representation is provided.

Dataset Specific Challenges: The fundamental difficulty lies in the structural divergence of the data formats, requiring a single neural solver to operate on all three representations despite their differing node counts, connectivity, and geometric structure, without a fixed grid resolution or topology to rely on. While the grid data benefits from spatial translation invariance and structured convolutions, the scattered mesh and Lagrangian particles lack a fixed global topology. Compounding this, the three representations encode the same physics, so the solver must distill a single consistent set of dynamics from all of them rather than fitting each representation independently. Crafting a unified latent space that bridges mesh-based and particle-based descriptions of the same flow, while maintaining accuracy across their disparate sampling densities, is the central hurdle for current models.

Dataset Generation: The dataset is constructed in two steps: a native Lagrangian dataset is generated first, and the Eulerian representations are then obtained from it by kernel interpolation. In the first step, a Smoothed Particle Hydrodynamics (SPH) (Gingold & Monaghan, 1977) simulation is run in PySPH (Ramachandran et al., 2021) for two-dimensional incompressible flow past a periodic array of cylinders: a cylinder of radius $a = 0.02$ sits at the centre of a channel of length $L = 0.12$ and height $H = 4a$ that is made periodic in the streamwise direction, so that it represents an infinite periodic array, and the flow is driven through it by a constant streamwise body force f_x. The fluid is discretized into roughly 16,000 Lagrangian particles—together with solid particles forming the cylinder and channel walls—and advanced with the Transport-Velocity Formulation (TVF) scheme, a weakly-compressible SPH method in which incompressibility is approximated through the artificial equation of state $p = c_0^2\rho$, the kinematic viscosity $\nu = 10^{-4}$ fixes the Reynolds number, and the time step is adaptively limited by acoustic (CFL), viscous, and body-force constraints; this yields the particle trajectories that form the Lagrangian representation, recording each particle's position and velocity as it advects through the domain. In the second step, the structured 64×32 grid and the unstructured 2048-point scattered mesh are produced from these particles by SPH kernel interpolation, which is the standard means of reconstructing field values in SPH rather than an ad-hoc resampling: any continuous field A is estimated at an arbitrary location $\mathbf{x}$ by the kernel-weighted particle sum

$$A(\mathbf{x}) = \sum_j \frac{m_j}{\rho_j} A_j \, W(\mathbf{x} - \mathbf{x}_j, \, h) \, , \tag{39}$$

where m_j, ρ_j, and A_j are the mass, density, and field value of particle j, and W is a smoothing kernel (a quintic spline) with smoothing length h. Evaluating this interpolant with $A = \mathbf{u}$ at the fixed grid nodes and mesh points gives the velocity field on each Eulerian representation directly from the same particles, so that all three representations are drawn from a single underlying simulation.

Governing Physics: The flow is governed by the incompressible Navier–Stokes equations, written here in Lagrangian (material) form, since the SPH discretization follows the fluid as a set of moving particles. Let a fluid particle occupy position $\mathbf{r}(t)$ and move with velocity $\mathbf{u}(t) = \dot{\mathbf{r}}(t)$; along particle paths the material derivative is $\frac{d}{dt} = \partial_t + \mathbf{u} \cdot \nabla$. The particle kinematics, mass conservation, and momentum balance then read

$$\frac{d\mathbf{r}}{dt} = \mathbf{u}, \tag{40}$$

$$\frac{d\rho}{dt} = -\rho \nabla \cdot \mathbf{u}, \tag{41}$$

$$\frac{d\mathbf{u}}{dt} = -\frac{1}{\rho}\nabla p + \nu \nabla^2 \mathbf{u} + \mathbf{f}, \tag{42}$$

where ρ is the density, p the pressure, ν the kinematic viscosity, and $\mathbf{f} = (f_x, 0)$ the constant streamwise body force (per unit mass) that drives the flow through the array. In the strictly incompressible limit these are closed by the divergence-free constraint $\nabla \cdot \mathbf{u} = 0$, so that $\frac{d\rho}{dt} = 0$; the weakly-compressible SPH model instead retains the continuity equation above and closes the system with the stiff barotropic equation of state

$$p = c_0^2 \rho, \tag{43}$$

in which the artificial sound speed c_0 is chosen large enough that density departures from the reference ρ_0 remain small and the incompressible limit is approached. The equations hold on the fluid domain Ω_f—the channel $[0, L] \times [0, H]$ with the cylinder disk of radius a removed—subject to no-slip conditions $\mathbf{u} = \mathbf{0}$ on the cylinder surface and the channel walls, together with streamwise periodicity of the fields, $\mathbf{u}(x + L, y, t) = \mathbf{u}(x, y, t)$ and $p(x + L, y, t) = p(x, y, t)$.

Experiment Results: This dataset comprises three distinct representations: an Eulerian grid, an Eulerian mesh, and Lagrangian particles. We use the averaged RMSE across all three representations as the primary evaluation metric, while also reporting performance on each individually. The complete results are presented in Table 21. Notably, all models exhibit poor performance on the Lagrangian representation. This suggests that the models prioritize the Eulerian data and fail to effectively embed the Lagrangian features. In this specific context, the automated research agent performs comparably to the general baselines.

	RMSE	RMSE (grid)	RMSE (mesh)	RMSE (particle)
GNOT	1.784×10^{-3}	4.250×10^{-5}	4.150×10^{-5}	1.999×10^{-3}
Transolver	$\mathbf{1.739 \times 10^{-3}}$	$\mathbf{3.350 \times 10^{-5}}$	3.560×10^{-5}	$\mathbf{1.947 \times 10^{-3}}$
Transolver++	1.775×10^{-3}	3.830×10^{-5}	$\mathbf{3.500 \times 10^{-5}}$	1.988×10^{-3}
LNO	1.785×10^{-3}	3.650×10^{-5}	3.650×10^{-5}	1.999×10^{-3}
AMG	6.491×10^{-2}	2.583×10^{-2}	9.078×10^{-2}	6.445×10^{-2}
Agent	1.785×10^{-3}	3.640×10^{-5}	3.640×10^{-5}	1.999×10^{-3}

Table 21: Performance comparison on the **Partial** dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 12 for visualizations of the dataset.

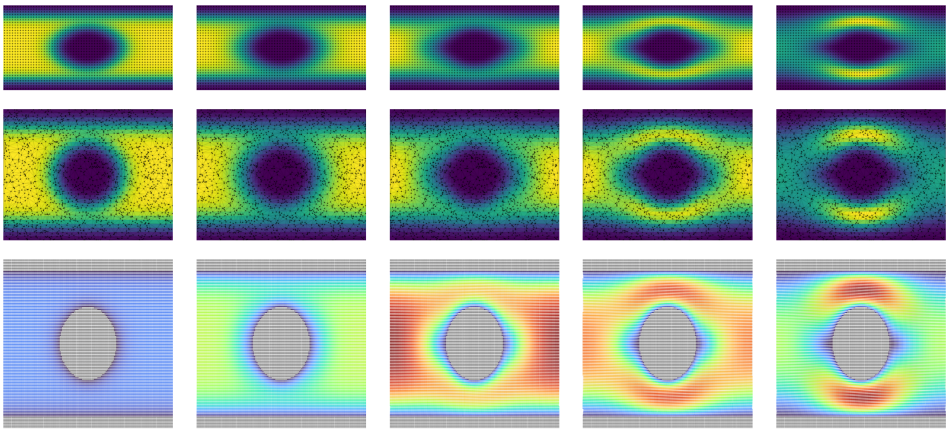


Figure 12: Visualization of the **MixedDomain** dataset. Rows from top to bottom display grid (Eulerian), mesh (Eulerian), particle (Lagrangian) representations, respectively.

I.7 MULTIPHYSICS

Dataset Description: This dataset is drawn from a coupled multiphysics simulation in which three physical subsystems continuously feed back on one another, following the coupling benchmark of Zhang et al. (2025). The source field is a scalar field produced by an internal generation process; it depends on the solid temperature through a feedback mechanism, on the fluid state, and on the control (boundary) parameters. The solid field is the temperature field inside a solid domain, driven by the heat source from the source field and cooled at its interface with the surrounding fluid. The fluid field is the fluid state that carries heat away from the solid, described by several channels (temperature, a velocity / momentum-like quantity, and further hydraulic variables) and driven by the heat flux leaving the solid surface. The three subsystems are solved by separate physics codes and coupled through a fixed-point (Picard) iteration: each solver is run with the current estimate of the other fields as its boundary/source condition, and the process repeats until all three fields stop changing.

Modeling Objective: We learn one neural surrogate per field. Each field is stored as a 5-D tensor of shape $[B, C, D, H, W]$: a batch of B samples, C physical channels, and a three-axis spatial grid (D, H, W). Let $x \in \mathbb{R}^{C_{\text{in}} \times D \times H \times W}$ denote a field's input conditioning tensor and $\hat{y} \in \mathbb{R}^{C_{\text{out}} \times D \times H \times W}$ its predicted solution. For each field we develop a parameterized surrogate Ψ_{field} that maps the conditioning fields driving it to that field's converged solution, $\hat{y} = \Psi_{\text{field}}(x)$, replacing one call to the corresponding legacy physics solver. The three fields differ in their channel roles: the source surrogate is conditioned on the boundary conditions together with the other fields and predicts the source field ($C_{\text{in}} = 2 \to C_{\text{out}} = 1$); the solid surrogate is conditioned on the source field and the fluid interface state and predicts the solid temperature ($C_{\text{in}} = 2 \to C_{\text{out}} = 1$); and the fluid surrogate is conditioned on the heat flux from the solid and predicts the 4-channel fluid state ($C_{\text{in}} = 1 \to C_{\text{out}} = 4$). Each surrogate is trained and validated on decoupled per-field $(\text{input}, \text{target})$ pairs, in which the target is that field's converged solution computed by the reference solver in isolation, and the reported score for a run is that field's validation accuracy on held-out decoupled data.

Dataset Specific Challenges: The central difficulty is that each surrogate must learn the conditional distribution of one field given the others, and these conditionals are tightly coupled: the source field is conditioned on the solid and fluid states, the solid on the source and fluid, and the fluid on the solid, so the fields are bound together through their mutual dependencies rather than being independently predictable. A surrogate is therefore useful only if it is accurate as a conditional in isolation and composes stably when its imperfect samples are fed back in as another field's conditioning input over many iterations.

Dataset Generation: We adopt the coupled three-field dataset from Zhang et al. (2025), generated with the finite-element and finite-volume multiphysics frameworks on a two-dimensional cell geometry partitioned into adjacent solid and fluid subdomains sharing a common interface. The source field is solved on a $[64, 20]$ mesh spanning the full domain, the solid temperature on a $[64, 8]$ mesh over the solid subdomain, and the fluid state on a $[64, 12]$ mesh over the fluid subdomain; the source and solid fields are evaluated at mesh points by the finite-element method, while the fluid field is evaluated at cell centers by the finite-volume method, with interpolation used to transfer values between the differing grids. The time step is adaptively controlled by the solver, and each trajectory is stored at 16 output steps, so the full spatio-temporal tensors have shapes $[b, 16, 64, 20]$, $[b, 16, 64, 8]$, and $[b, 16, 64, 12]$ for the three fields. Two kinds of data are produced. The decoupled data used for training and validation are generated by a pre-iteration procedure: each field is computed in isolation while the others are held at assumed conditions, sweeping the boundary condition imposed on the source field, so that each converged single-field solution is paired with the conditioning fields that drove it; the most expensive field (the fluid) is excluded from pre-iteration to reduce cost, and the remaining fields are computed in sequence. The coupled ground truth used only for the final test is generated with an operator-splitting iterative algorithm that exchanges information between the three fields at every time step and iterates to convergence.

Governing Physics: The dataset is governed by three coupled fields on the domain $x \in [0, L_s + L_f]$, $y \in [0, L_y]$, $t \in [0, 5]$: a source field ϕ obeying a diffusion–reaction balance, a solid temperature T_s obeying heat conduction with a volumetric source, and a fluid state $(\vec{u}, p, T_f)$ obeying the incom-

pressible flow and convective heat-transport equations. The source field evolves as

$$\frac{1}{v}\frac{\partial \phi}{\partial t} = D\,\Delta\phi + \big(\nu\Sigma_f - \Sigma_a(T)\big)\phi, \qquad x \in [0, L_s + L_f], \tag{44}$$

with D a diffusion coefficient, $\nu\Sigma_f$ a generation term, and $\Sigma_a(T)$ an absorption/removal term that depends on the local temperature; a time-varying inflow condition $\phi(0, y, t) = f(y, t)$ is imposed on the left edge and homogeneous conditions on the others. This temperature dependence of $\Sigma_a(T)$ is the negative feedback that ties the source field back to the thermal state. The solid temperature is driven by the source field through a heat-generation term $A\phi_s$,

$$\rho c_p \frac{\partial T_s}{\partial t} = \nabla \cdot \big(k_s \nabla T_s\big) + A\,\phi_s, \qquad x \in [0, L_s], \tag{45}$$

with insulated conditions on the top and bottom edges. The fluid subdomain satisfies the incompressible Navier–Stokes equations together with convective heat transport,

$$\nabla \cdot \vec{u} = 0, \qquad \rho\left(\frac{\partial \vec{u}}{\partial t} + \vec{u}\cdot\nabla\vec{u}\right) = -\nabla p + \mu\nabla^2\vec{u} + \vec{f}, \tag{46}$$

$$\rho c_p\left(\frac{\partial T_f}{\partial t} + \vec{u}\cdot\nabla T_f\right) = k_f \nabla^2 T_f, \qquad x \in [L_s, L_s + L_f], \tag{47}$$

where ρ is density, μ dynamic viscosity, c_p specific heat, and k_s, k_f the solid and fluid conductivities (both functions of temperature). The solid and fluid fields are tied together at their shared interface $x = L_s$ by continuity of temperature and of heat flux,

$$T_s(L_s, y, t) = T_f(L_s, y, t), \qquad k_f \frac{\partial T_f(L_s, y, t)}{\partial x} = k_s \frac{\partial T_s(L_s, y, t)}{\partial x}, \tag{48}$$

which is precisely the strong interface coupling the surrogate must respect: the source field feeds heat into the solid, the solid exchanges heat with the fluid across the interface, and the fluid temperature feeds back into the source field through the temperature-dependent removal term $\Sigma_a(T)$.

Experiment Results: The complete experimental results are presented in Table 22.

	RMSE (source)	FRMSE (source)	RMSE (solid)
GNOT	1.710×10^{-2}	4.983×10^{-1}	1.212×10^{-1}
Transolver	1.870×10^{-1}	6.146×10^{0}	8.238×10^{-2}
Transolver++	1.954×10^{-1}	6.490×10^{0}	7.526×10^{-2}
LNO	3.315×10^{-1}	5.753×10^{0}	7.282×10^{-1}
AMG	7.817×10^{-3}	2.438×10^{-1}	1.125×10^{-1}
Agent	$\mathbf{6.585 \times 10^{-3}}$	$\mathbf{2.061 \times 10^{-1}}$	$\mathbf{3.802 \times 10^{-2}}$

	FRMSE (solid)	RMSE (fluid)	FRMSE (fluid)
GNOT	2.446×10^{0}	6.879×10^{-2}	$\mathbf{1.826 \times 10^{0}}$
Transolver	1.843×10^{0}	1.201×10^{-1}	3.327×10^{0}
Transolver++	1.697×10^{0}	1.330×10^{-1}	3.686×10^{0}
LNO	1.648×10^{1}	4.316×10^{-1}	5.735×10^{0}
AMG	2.447×10^{0}	$\mathbf{9.876 \times 10^{-2}}$	2.603×10^{0}
Agent	$\mathbf{7.926 \times 10^{-1}}$	1.096×10^{-1}	2.947×10^{0}

Table 22: Performance comparison on the **MultiPhysics** dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 13 for visualizations of the dataset.

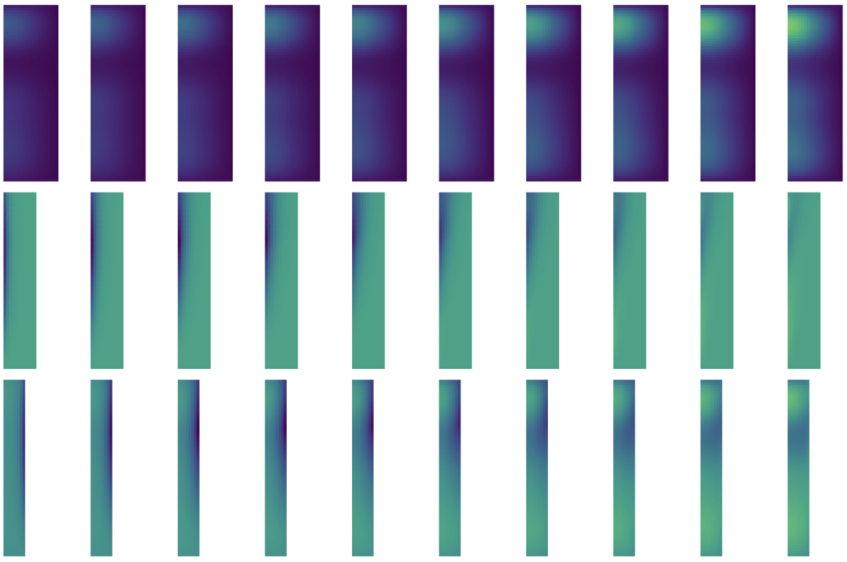


Figure 13: Visualization of the **MultiPhysics** dataset. Rows from top to bottom display source, fluid, and solid, respectively.

I.8 3D

Dataset Description: This dataset consists of three-dimensional, isothermal Magnetohydrodynamic (MHD) turbulence. It captures the complex interplay between magnetic fields and conductive fluid dynamics, essential processes for modeling the solar wind, galaxy formation, and Interstellar Medium (ISM) dynamics. The simulation represents a periodic cubic domain without self-gravity.

Modeling Objective: The physical state at time t is represented by a high-dimensional tensor $x_t \in \mathbb{R}^{7\times64\times64\times64}$. This 7-channel input encapsulates the scalar density field ρ, the velocity vector field $\mathbf{u} = (u_x, u_y, u_z)$, and the magnetic vector field $\mathbf{B} = (B_x, B_y, B_z)$. The objective is to develop a neural simulator, Ψ, capable of next-step prediction, mapping the current volumetric state to the full state at the subsequent timestep: $\hat{x}_{t+1} = \Psi(x_t)$.

Dataset Specific Challenges: The transition from 2D to 3D PDE modeling introduces a curse of dimensionality that complicates both accuracy and computational feasibility. At a resolution of 64^3 with 7 input channels, a single snapshot contains over 1.8 million data points, creating a massive memory footprint that often exceeds the VRAM capacity of standard hardware. This scale makes existing neural architectures prohibitively slow to train and difficult to deploy for real-time inference. Beyond the computational load, modeling the 3D spatial relationships is inherently more complex than in 2D counterparts.

Dataset Generation: We directly adopt the dataset from Ohana et al. (2024). The dataset is generated by simulating isothermal, compressible magnetohydrodynamic (MHD) turbulence without self-gravity, representative of the diffuse interstellar medium, using a second-order-accurate finite-difference code that solves the ideal MHD equations on a triply-periodic uniform Cartesian grid. The flow is initialized from a uniform density field of unit mean and a uniform magnetic field oriented along a single axis, and is driven at large scales by a random, solenoidal (divergence-free) forcing that injects energy near wavenumber $k \approx 2.5$, stirring the gas until a statistically steady turbulent cascade develops. Because the equation of state is isothermal, each trajectory is fully characterized by two dimensionless control parameters: the sonic Mach number $\mathcal{M}_s = \langle|\mathbf{v}|/c_s\rangle$, which sets the compressibility of the flow, and the Alfvénic Mach number $\mathcal{M}_A = \langle|\mathbf{v}|/v_A\rangle$ with Alfvén speed $v_A = |\mathbf{B}|/\sqrt{\rho}$, which sets the dynamical importance of the magnetic field. The dataset sweeps all combinations of $\mathcal{M}_s \in \{0.5, 0.7, 1.5, 2.0, 7.0\}$ and $\mathcal{M}_A \in \{0.7, 2.0\}$, spanning subsonic to strongly supersonic and sub-Alfvénic to super-Alfvénic regimes, with ten random initial conditions per parameter pair for a total of 100 trajectories. Each simulation is first computed at a resolution of 256^3 and then downsampled to 64^3 after anti-aliasing with an ideal low-pass filter; snapshots of the density, velocity, and magnetic field are written at fixed intervals $\Delta t = 0.01$ in dimensionless units over the range $t \in [0, 1]$, yielding 100 temporal frames per trajectory.

Governing Physics: The dataset models the dynamics of an electrically conducting, compressible, isothermal fluid coupled to a magnetic field, governed by the equations of ideal magnetohydrodynamics on the triply-periodic domain $\Omega = [0, L_x] \times [0, L_y] \times [0, L_z]$,

$$\frac{\partial\rho}{\partial t}+\nabla\cdot(\rho\mathbf{v}) = 0, \qquad \frac{\partial(\rho\mathbf{v})}{\partial t}+\nabla\cdot\big(\rho\,\mathbf{v}\otimes\mathbf{v}-\mathbf{B}\otimes\mathbf{B}\big)+\nabla p = \mathbf{0}, \qquad \frac{\partial\mathbf{B}}{\partial t}-\nabla\times(\mathbf{v}\times\mathbf{B}) = \mathbf{0}, \tag{49}$$

where ρ is the density, $\mathbf{v}$ the velocity, $\mathbf{B}$ the magnetic field (absorbing a factor of $1/\sqrt{4\pi}$ so that magnetic pressure appears directly), and p the gas pressure. The three relations express, respectively, the conservation of mass through the continuity equation, the conservation of momentum with the combined gas and Maxwell stresses $-\mathbf{B}\otimes\mathbf{B}$ carrying the magnetic tension and pressure, and the ideal induction equation, which advects the magnetic field with the flow and enforces the frozen-in condition of a perfectly conducting medium. The magnetic field additionally satisfies the solenoidal constraint

$$\nabla\cdot\mathbf{B} = 0, \tag{50}$$

which is preserved for all time given divergence-free initial data. The system is closed by an isothermal equation of state

$$p = c_s^2\,\rho, \tag{51}$$

with constant sound speed c_s, so that no separate energy equation is required and the thermal pressure is set entirely by the local density. Under this closure the flow admits no intrinsic length or time scale, and its statistics are controlled solely by the sonic and Alfvénic Mach numbers,

$$\mathcal{M}_s = \frac{\langle|\mathbf{v}|\rangle}{c_s}, \qquad \mathcal{M}_A = \frac{\langle|\mathbf{v}|\rangle}{v_A}, \qquad v_A = \frac{|\mathbf{B}|}{\sqrt{\rho}}, \tag{52}$$

together with the plasma-β parameter $\beta = 2\,(\mathcal{M}_A/\mathcal{M}_s)^2$ measuring the ratio of thermal to magnetic pressure. The interplay of compressibility and magnetization gives rise to a rich family of behaviors, from nearly incompressible, magnetically dominated turbulence at low $\mathcal{M}_s$ and low $\mathcal{M}_A$ to highly compressible, shock-dominated flows with sharp density contrasts at high $\mathcal{M}_s$.

Experiment Results: The complete results are presented in Table 23. Because this is Eulerian grid data, we evaluate performance using RMSE, FRMSE, and Kinetic Energy. We also incorporate a dataset-specific metric, Magnetic Energy, to compute how well the model conserves the magnetic field properties over time. Experimental results demonstrate that the agent's design achieves substantially better performance. AMG is unsuitable for this application because its graph architecture scales poorly to large 3D datasets, resulting in prohibitive training times.

	RMSE	FRMSE	Kinetic Energy	Magnetic Energy
GNOT	2.230×10^{-1}	7.444×10^{1}	7.545×10^{-2}	6.777×10^{-2}
Transolver	2.279×10^{-1}	7.532×10^{1}	7.572×10^{-2}	6.807×10^{-2}
Transolver++	2.257×10^{-1}	7.470×10^{1}	7.592×10^{-2}	6.801×10^{-2}
LNO	2.238×10^{-1}	7.469×10^{1}	7.611×10^{-2}	6.821×10^{-2}
AMG	–	–	–	–
Agent	$\mathbf{7.182 \times 10^{-2}}$	$\mathbf{2.577 \times 10^{1}}$	$\mathbf{3.313 \times 10^{-2}}$	$\mathbf{2.317 \times 10^{-2}}$

Table 23: Performance comparison on the 3D dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 14 for visualizations of the dataset.

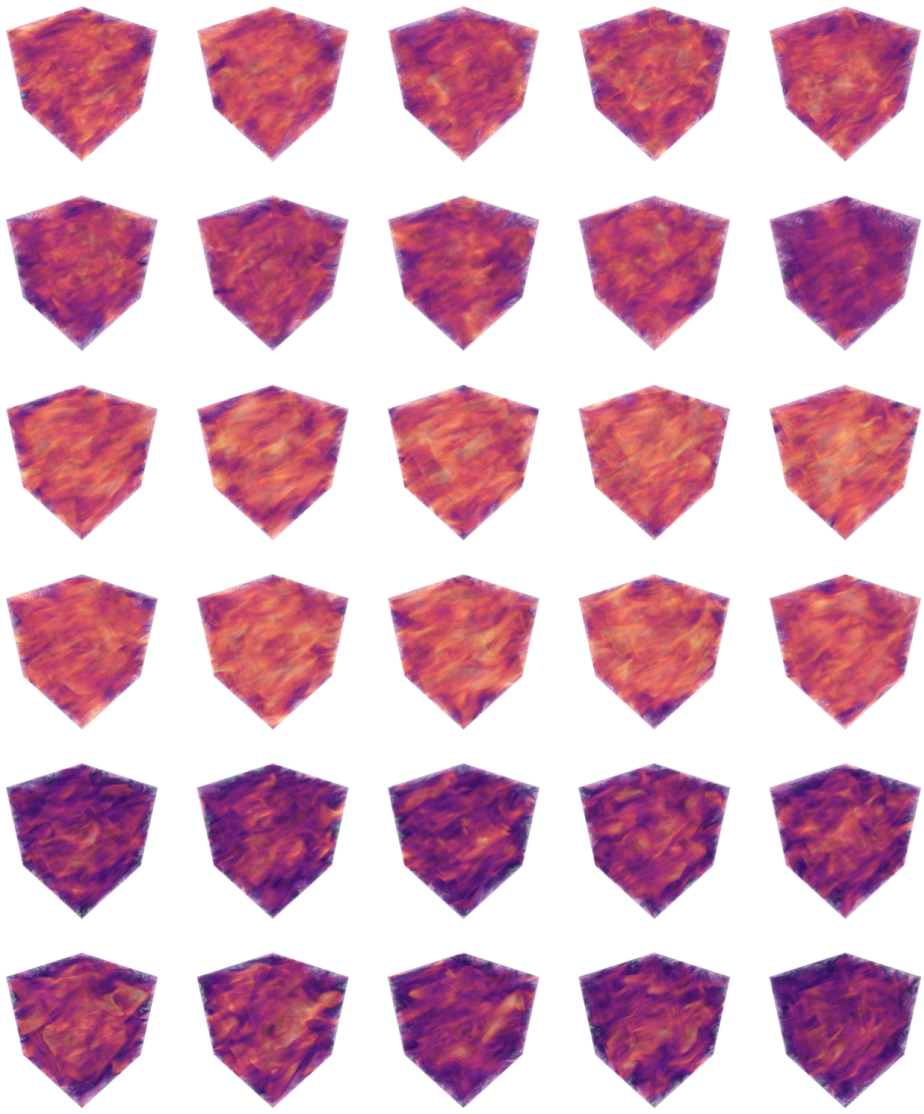


Figure 14: Visualization of the 3D dataset. From top to bottom, the first two rows display density, the middle two the magnetic field norm, and the last two the velocity norm.

I.9 IRREGULARTIME

Dataset Description: This dataset captures two-dimensional incompressible Navier-Stokes dynamics sampled at irregular time intervals. The flow evolves on a 128×128 spatial grid under the interplay of nonlinear advection and viscous diffusion. Unlike standard benchmarks recorded on a uniform temporal grid, consecutive snapshots within each trajectory are separated by varying time gaps, reflecting practical settings where data acquisition is asynchronous or adaptively sampled.

Modeling Objective: The state of the fluid system is represented by the tensor $x_t \in \mathbb{R}^{1\times128\times128}$, where the single feature channel corresponds to the vorticity field evaluated across a 128×128 spatial discretization grid. The system is explicitly conditioned on a scalar time increment $dt \in \mathbb{R}^1$, specifying the temporal gap to the next snapshot. The objective is to develop a time-conditioned neural simulator, defined as a function Ψ, that maps the current system state and the elapsed time increment to the state after that interval. Formally, we aim to predict the next state $\hat{x}_{t+dt} = \Psi(x_t, dt)$, where $\hat{x}_{t+dt} \in \mathbb{R}^{1\times128\times128}$ matches the shape of the input state.

Dataset Specific Challenges: The dataset's primary challenge lies in the architectural design required to condition the neural model on a continuous temporal increment. Standard neural operators are trained to advance the system by a single fixed time step, implicitly baking the step size into their learned dynamics; here, the model must instead condition explicitly on dt and generalize across a continuous range of temporal gaps. Furthermore, the effective difficulty of each prediction varies with dt: larger gaps require capturing more accumulated nonlinear evolution in a single forward pass, while smaller gaps demand near-identity behavior, and a single model must remain accurate and temporally stable across this entire spectrum.

Dataset Generation: The dataset is generated following the procedure outlined in Section I.5, where recording times are randomly sampled.

Governing Physics: The physical setup follows the description in Section I.5, governed by the 2D incompressible Navier-Stokes equations in vorticity form.

Experiment Results: Table 24 presents the complete experimental results. We use standard metrics for Eulerian datasets to capture both pointwise error and physical consistency. The agent's design consistently achieves superior performance across all evaluated metrics.

	RMSE	FRMSE	Kinetic Energy
GNOT	2.577×10^{0}	2.165×10^{2}	9.050×10^{-1}
Transolver	2.623×10^{0}	2.092×10^{2}	1.376×10^{0}
Transolver++	2.621×10^{0}	2.100×10^{2}	1.389×10^{0}
LNO	2.622×10^{0}	2.124×10^{2}	1.431×10^{0}
AMG	2.415×10^{0}	2.084×10^{2}	7.990×10^{-1}
Agent	$\mathbf{6.511 \times 10^{-1}}$	$\mathbf{6.022 \times 10^{1}}$	$\mathbf{9.200 \times 10^{-2}}$

Table 24: Performance comparison on the **IrregularTime** dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 15 for visualizations of the dataset.

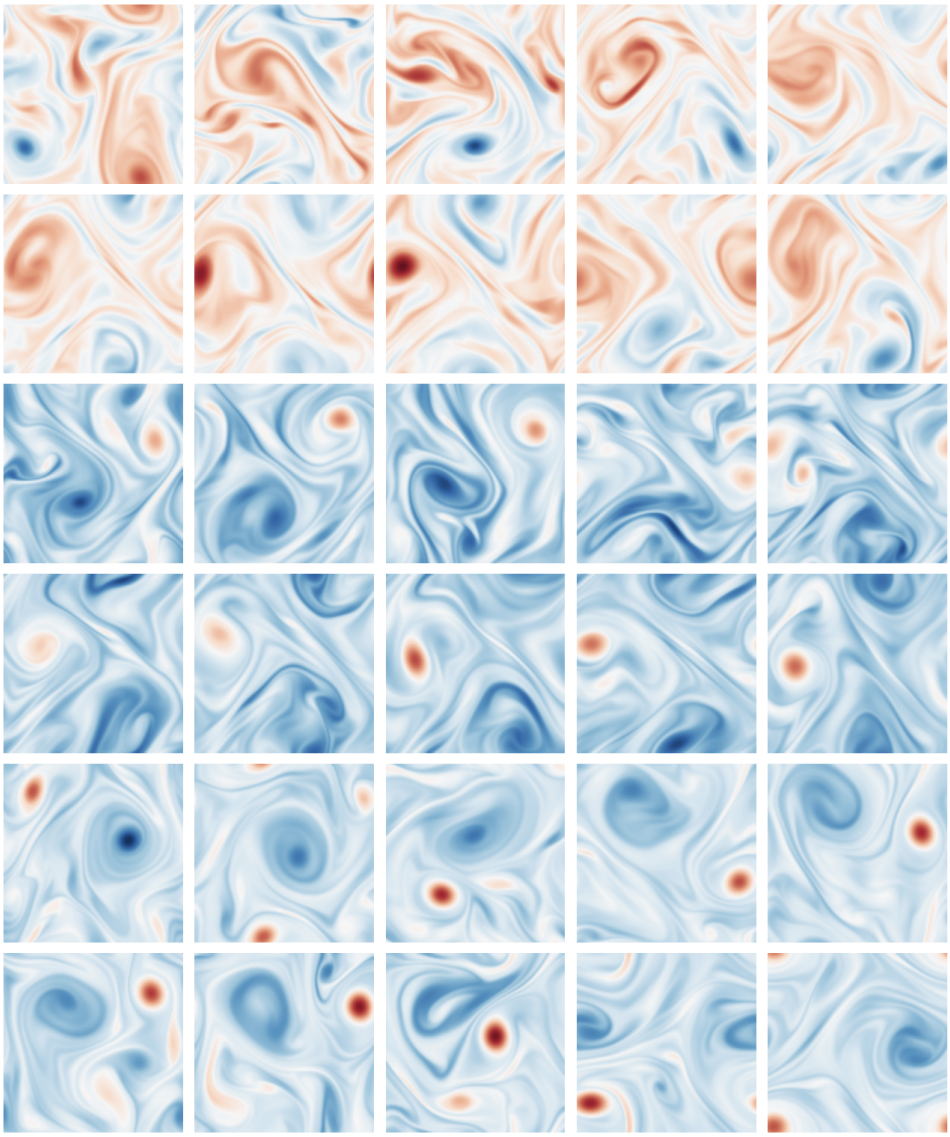


Figure 15: Visualization of the **`IrregularTime`** dataset. The dataset contains solutions to the Navier-Stokes equations in vorticity form sampled at irregular time intervals.

I.10 LAGINLETOUTLET

Dataset Description: This dataset contains Lagrangian particle simulations of jet engine flow in an open domain with an inlet and an outlet. The fluid is represented by a time-evolving set of particles carrying position and velocity, with new particles continuously injected at the inlet and existing particles removed once they exit through the outlet. Unlike standard Lagrangian benchmarks, where a fixed set of particles persists for the entire trajectory and the model can be supervised through a one-to-one correspondence between input and output particles, the particle set here changes identity and cardinality between consecutive time steps, so no such correspondence exists.

Modeling Objective: The state of the fluid system is represented by the tensor $x_t \in \mathbb{R}^{N\times 4}$, where each of the N particles carries its two-dimensional position and velocity. The model is additionally conditioned on a set of query positions $Q_{t+1} \in \mathbb{R}^{M\times 2}$ at the subsequent time step. The objective is to develop a neural simulator, defined as a function Ψ, that encodes the input particle set into a latent representation of the flow field and decodes the fluid velocity at each query location. Formally, we aim to predict $\hat{v}_{t+1} = \Psi(x_t, Q_{t+1})$, where $\hat{v}_{t+1} \in \mathbb{R}^{M\times 2}$ is the predicted velocity at the query positions, rather than the advected state of a fixed set of tracked particles. A validity mask over the input particles supports batching across samples with differing particle counts by excluding padded entries from the encoding.

Dataset Specific Challenges: The dataset's primary challenge lies in the absence of particle correspondence, which removes the standard supervision signal of per-particle displacement targets. The model cannot learn a map from each input particle to its own future state, and must instead treat the particles as an unstructured, permutation-invariant sampling of an underlying velocity field. This field is queried at positions that are decoupled from the input set, requiring the model to represent the continuous flow rather than track discrete entities across time steps.

Dataset Generation: The dataset is generated by simulating a planar high-speed liquid jet impinging on a flat wall with weakly-compressible Smoothed Particle Hydrodynamics (SPH) (Zhan et al., 2025; Domínguez et al., 2022), in which the fluid is discretized into Lagrangian particles advected through a fixed inlet/outlet buffer configuration rather than solved on a grid. The two-dimensional domain is centered on the origin: a thin fluid strip of width $30\,\mathrm{mm}$ and thickness $5\,\mathrm{mm}$ is placed at height $z = 0.12\,\mathrm{m}$ and driven downward through a top inlet zone that continuously injects water at a fixed velocity $v_{\mathrm{in}} = 20\,\mathrm{m/s}$, while the initial fluid block is likewise assigned a downward velocity $-v_{\mathrm{in}}$. The jet falls under gravity $g_z = -9.81\,\mathrm{m/s^2}$ and impinges on a solid bottom wall built from four boundary layers spanning the base of the domain, spreading laterally and leaving through two open outlets on the left and right sides, each realized as a buffer zone with velocity and density extrapolated from the interior ghost nodes so that fluid exits without spurious reflection. The particles are laid out at an initial inter-particle spacing $\Delta p = 2\,\mathrm{mm}$, with smoothing length $h = \Delta p\sqrt{3}$ and a reference density $\rho_0 = 1000\,\mathrm{kg/m^3}$. Incompressibility is approximated through a Tait equation of state $p = \frac{c_0^2\rho_0}{\gamma}\left[(\rho/\rho_0)^\gamma - 1\right]$ with polytropic exponent $\gamma = 7$ and an artificial sound speed $c_0 = 230\,\mathrm{m/s}$, chosen large enough relative to the jet velocity that density fluctuations remain small. Solid boundaries use the modified Dynamic Boundary Conditions (mDBC), in which boundary-particle densities are reconstructed by extrapolation from ghost nodes placed in the fluid to reduce the unphysical fluid–wall gap of the standard scheme; viscous stresses are modeled with a laminar viscosity $\nu = 10^{-6}\,\mathrm{m^2/s}$ combined with a Sub-Particle-Scale (SPS) turbulence closure. Time integration uses a Symplectic scheme with a Wendland kernel and an adaptive step set by a CFL condition. The simulation integrates to a final time $t_f = 12\,\mathrm{s}$, writing particle snapshots every $\Delta t_{\mathrm{out}} = 0.001\,\mathrm{s}$.

Governing Physics: The dataset models the flow of a single-phase, weakly-compressible, viscous liquid injected as a jet and impinging on a solid wall, governed by the Lagrangian form of the continuity and Navier–Stokes equations on the two-dimensional domain Ω under the body force $\mathbf{g} = (0, g_z)$,

$$\frac{D\rho}{Dt} = -\rho\,\nabla\cdot\mathbf{u}, \qquad \frac{D\mathbf{u}}{Dt} = -\frac{1}{\rho}\nabla p + \nu\,\nabla^2\mathbf{u} + \mathbf{g}, \tag{53}$$

where $\frac{D}{Dt} = \partial_t + \mathbf{u}\cdot\nabla$ is the material derivative following a fluid particle, $\mathbf{u}$ the velocity, p the pressure, ρ the density, and ν the kinematic viscosity. Rather than enforcing strict incompressibility,

the weakly-compressible formulation closes the system with the barotropic Tait equation of state

$$p = \frac{c_0^2\,\rho_0}{\gamma}\left[\left(\frac{\rho}{\rho_0}\right)^{\gamma} - 1\right], \qquad \gamma = 7, \tag{54}$$

which relates pressure to density through the reference density ρ_0 and an artificial sound speed c_0. The sound speed is set well above the maximum expected flow speed so that the Mach number $\mathrm{Ma} = U/c_0$ remains small and the resulting density variations, of order Ma^2, stay within a few percent, recovering the incompressible limit while permitting an explicit time-stepping scheme. The jet enters through a fixed-velocity inlet condition $\mathbf{u} = (0, -v_{\mathrm{in}})$ and leaves through open outlets where the velocity and density are extrapolated from the interior, while a no-penetration condition is enforced at the solid bottom wall through the modified Dynamic Boundary Conditions. The dynamics are characterized by the Reynolds number and the Froude number,

$$\mathrm{Re} = \frac{v_{\mathrm{in}}\,d}{\nu}, \qquad \mathrm{Fr} = \frac{v_{\mathrm{in}}}{\sqrt{|g_z|\,d}}, \qquad \mathrm{Ma} = \frac{v_{\mathrm{in}}}{c_0}, \tag{55}$$

where d denotes the jet thickness. With a high injection velocity and a thin jet, the flow is strongly inertia-dominated, so the impingement produces a thin spreading wall layer, lateral jetting toward the outlets, and unsteady breakup at the free surface where the fluid separates from the wall.

Experiment Results: Table 25 presents the complete results. In addition to pointwise RMSE, we introduce LagKE to evaluate physical consistency, alongside EuARMSE and EuAFRMSE to assess practical utility. We observe that while the agent achieves strong pointwise accuracy, it struggles to maintain physical consistency. Furthermore, standard transformer and graph-based models are inapplicable here because the dataset features differing numbers of input and output particles. Instead, LNO condenses the initial conditions into a fixed-size latent representation and applies the processor at this level, while the decoder allows for position-based querying.

	RMSE	LagKE	EuARMSE	EuAFRMSE
GNOT	–	–	–	–
Transolver	–	–	–	–
Transolver++	–	–	–	–
LNO	6.105×10^{-1}	$\mathbf{1.349 \times 10^{3}}$	1.129×10^{-1}	2.621×10^{0}
AMG	–	–	–	–
Agent	$\mathbf{5.530 \times 10^{-1}}$	1.662×10^{3}	$\mathbf{9.951 \times 10^{-2}}$	$\mathbf{2.299 \times 10^{0}}$

Table 25: Performance comparison on the **`LagInletOutlet`** dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 16 for visualizations of the dataset.

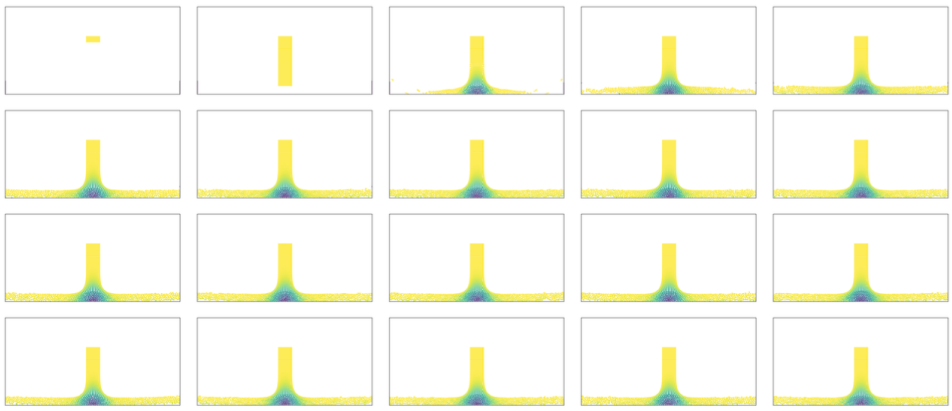


Figure 16: Visualization of the **LagInletOutlet** dataset. The dataset uses Lagrangian particles to simulate high-speed exhaust from a jet engine. The airflow enters the computational domain from the top boundary, impinges on the bottom wall, and subsequently exits through the left and right boundaries.

I.11 LAGFSI

Dataset Description: This dataset contains two-dimensional Lagrangian particle simulations of fluid-structure interaction, in which a rigid solid sphere falls into a body of water. Both phases are discretized as particles: fluid particles follow the free-surface flow, while solid particles collectively represent the sphere and must move as a coherent rigid body. Unlike single-phase Lagrangian benchmarks, the dynamics here are governed by two-way coupling: the falling solid displaces and accelerates the surrounding fluid, while the fluid exerts buoyancy, drag, and impact forces back on the solid.

Modeling Objective: The state of the system is represented by the tensor $x_t \in \mathbb{R}^{N\times 13}$, where each particle's feature vector concatenates its current position in $\mathbb{R}^2$, a history of its 5 most recent velocities in $\mathbb{R}^{10}$ obtained by finite-differencing consecutive positions over the preceding frames, and a particle type in $\{0, 1\}$ indicating solid or fluid. The model additionally receives a batch index vector $\boldsymbol{b} \in \{0, \ldots, B-1\}^N$ assigning each particle to its sample, since samples with differing particle counts are concatenated by a graph data loader rather than stacked along a batch dimension, and a particle type vector $p \in \{0, 1\}^{N\times 1}$, provided as a standalone input in addition to its copy inside the feature vector. The objective is to develop a neural simulator, defined as a function Ψ, such that $\hat{v}_{t+1} = \Psi(x_t, \boldsymbol{b}, p)$, where $\hat{v}_{t+1} \in \mathbb{R}^{N\times 2}$ is the predicted velocity over the next time step, from which particle positions are advanced during rollout.

Dataset Specific Challenges: The dataset presents two intertwined challenges. First, the two phases obey qualitatively different dynamics under two-way coupling: fluid particles respond to local pressure and viscous interactions with their neighbors, while solid particles are constrained by global rigid-body kinematics yet driven by contact forces from the surrounding fluid. A single model must capture both behaviors and the momentum exchange between them, including the impulsive interaction at impact, using only the particle type label to differentiate the phases. Second, the solid has no explicitly enforced rigidity: its shape is maintained only through the predicted per-particle velocities. Small inconsistencies among solid-particle predictions accumulate over autoregressive rollout into stretching, shearing, or fragmentation of the sphere, so the model must keep the solid particles' motion mutually consistent, effectively learning the rigidity constraint, while remaining accurate for the surrounding free-surface flow.

Dataset Generation: The dataset is generated by simulating a rigid sphere falling into a hydrostatic water tank with weakly-compressible Smoothed Particle Hydrodynamics (SPH), in which both the fluid and the solid are discretized into Lagrangian particles rather than solved on a fixed grid. The two-dimensional domain is a rectangular tank of width $L_x = 100\,\mathrm{mm}$, filled to a variable still-water height H and enclosed by solid boundary walls two particle-layers thick along the bottom and sides. A circular rigid body of radius R and density $\rho_s = 500\,\mathrm{kg/m^3}$ is initialized at a prescribed center (x_0, y_0) above or within the water and released from rest, falling under gravity $g_y = -9.81\,\mathrm{m/s^2}$. A defining feature of this dataset is that the sphere radius, its initial position, and the water height vary across trajectories, which induces a broad spectrum of impact and floatation regimes: spheres released higher above the surface strike with greater momentum and generate more violent splashing and cavity formation, whereas those initialized near or below the surface settle more gently toward their buoyant equilibrium. The fluid particles are laid out on a uniform lattice at spacing $\Delta x = 2\,\mathrm{mm}$ with smoothing length $h = 1.2\,\Delta x$ and are initialized at hydrostatic density through the Tait relation, while the sphere is discretized at a finer spacing to resolve its curved interface; the per-particle mass of each phase is set to reproduce its reference density. The fluid dynamics are advanced with a weakly-compressible SPH scheme, in which incompressibility is approximated through an artificial equation of state with polytropic exponent $\gamma = 7$ and an artificial sound speed $c_0 = 2\sqrt{2|g_y|H}$ chosen from the maximum expected flow speed, and momentum transport uses an artificial viscosity with coefficient $\alpha = 0.1$. The rigid body is integrated as a single solid whose particles move together under rigid-body motion, driven by the net force and torque accumulated from gravity and from the surrounding fluid; contact with the tank walls is handled by a penalty collision force with stiffness $k_n = 10^5$. Time integration uses a predictor–corrector (EPEC) scheme for the fluid and a Runge–Kutta rigid-body step for the solid, with a fixed step set by a CFL-type acoustic limit.

Governing Physics: The dataset models a two-way-coupled fluid–structure interaction problem, in which a weakly-compressible viscous fluid and a rigid solid body exchange momentum across their

shared interface. The fluid is governed by the Lagrangian continuity and momentum equations,

$$\frac{D\rho}{Dt} = -\rho\,\nabla\cdot\mathbf{u}, \qquad \frac{D\mathbf{u}}{Dt} = -\frac{1}{\rho}\nabla p + \nu\,\nabla^2\mathbf{u} + \mathbf{g} + \mathbf{f}_{s\to f}, \tag{56}$$

where $\frac{D}{Dt} = \partial_t + \mathbf{u}\cdot\nabla$ is the material derivative, $\mathbf{u}$ the fluid velocity, p the pressure, ρ the density, ν the kinematic viscosity, and $\mathbf{g} = (0, g_y)$ gravity. The system is closed by the barotropic Tait equation of state $p = \frac{c_0^2\rho_0}{\gamma}\left[(\rho/\rho_0)^\gamma - 1\right]$ with $\gamma = 7$, so that the artificial sound speed c_0 keeps density fluctuations small and recovers the incompressible limit. The solid is governed by the rigid-body equations of motion, in which its particles translate and rotate as a single body according to the Newton–Euler laws,

$$M\,\frac{d\mathbf{U}}{dt} = M\mathbf{g} + \mathbf{F}_{f\to s}, \qquad \mathbf{I}\,\frac{d\boldsymbol{\omega}}{dt} = \mathbf{T}_{f\to s}, \qquad \mathbf{u}_i = \mathbf{U} + \boldsymbol{\omega}\times(\mathbf{r}_i - \mathbf{r}_c), \tag{57}$$

where M is the body mass, $\mathbf{I}$ its moment of inertia, $\mathbf{U}$ and $\boldsymbol{\omega}$ the translational and angular velocities, $\mathbf{r}_c$ the center of mass, and $\mathbf{u}_i$ the velocity of constituent particle i at position $\mathbf{r}_i$. The two systems are coupled through the reciprocal interfacial forces at the boundary between the phases: the fluid exerts a hydrodynamic load, comprising buoyancy, drag, and impact pressure, on the solid,

$$\mathbf{F}_{f\to s} = \sum_{i\in s}\mathbf{f}^i_{f\to s}, \qquad \mathbf{T}_{f\to s} = \sum_{i\in s}(\mathbf{r}_i - \mathbf{r}_c)\times\mathbf{f}^i_{f\to s}, \tag{58}$$

while by Newton's third law the solid exerts an equal and opposite reaction $\mathbf{f}_{s\to f} = -\mathbf{f}_{f\to s}$ back on the fluid, displacing and accelerating it as the body moves. It is this bidirectional exchange — the solid pushing the fluid aside while the fluid buoys, retards, and deflects the solid — that produces the characteristic dynamics of the dataset: the impact and cavity formation as the sphere enters the water, the deceleration and rebound as it interacts with the free surface, and its eventual rise toward the buoyant equilibrium set by the density ratio $\rho_s/\rho_0 = 0.5$.

Experiment Results: We present the complete experimental results in Table 26, where we not only dissect the error into fluid and solid components for finer analysis but also report metrics for physical consistency and practical utility. The RMSE reported in the main table reflects the average across both the fluid and solid domains. We observe that the agent performs poorly across all evaluated metrics.

	RMSE (fluid)	LagKE (fluid)	Sinkhorn (fluid)	RMSE (solid)
GNOT	$\mathbf{9.505\times10^{-5}}$	1.043×10^{-4}	$\mathbf{3.336\times10^{-10}}$	6.235×10^{-5}
Transolver	9.731×10^{-5}	$\mathbf{7.354\times10^{-5}}$	3.551×10^{-10}	5.417×10^{-5}
Transolver++	1.001×10^{-4}	8.819×10^{-5}	3.617×10^{-10}	7.156×10^{-5}
LNO	1.008×10^{-4}	8.440×10^{-5}	6.053×10^{-10}	6.830×10^{-5}
AMG	9.554×10^{-5}	1.112×10^{-4}	4.462×10^{-10}	7.473×10^{-5}
Agent	4.024×10^{-4}	1.315×10^{-3}	1.269×10^{-8}	$\mathbf{4.071\times10^{-5}}$

	LagKE (solid)	Sinkhorn (solid)	EuARMSE (fluid)	EuAFRMSE (fluid)
GNOT	1.202×10^{-4}	3.890×10^{-9}	$\mathbf{1.029\times10^{-4}}$	$\mathbf{5.782\times10^{-3}}$
Transolver	4.306×10^{-5}	3.045×10^{-9}	1.189×10^{-4}	6.571×10^{-3}
Transolver++	9.993×10^{-5}	5.892×10^{-9}	1.156×10^{-4}	6.488×10^{-3}
LNO	6.977×10^{-5}	4.417×10^{-9}	1.270×10^{-4}	7.133×10^{-3}
AMG	1.034×10^{-4}	5.762×10^{-9}	1.051×10^{-4}	6.004×10^{-3}
Agent	$\mathbf{4.185\times10^{-5}}$	$\mathbf{2.208\times10^{-9}}$	5.658×10^{-4}	4.089×10^{-2}

Table 26: Performance comparison on the **LagFSI** dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 17 for visualizations of the dataset.

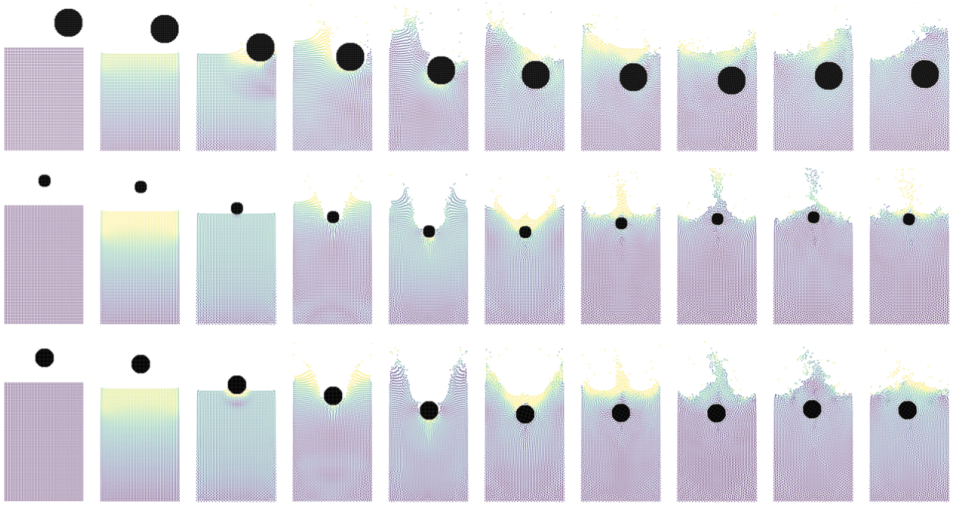


Figure 17: Visualization of the **LagFSI** dataset. The dataset simulates a solid sphere falling into a volume of water, with both phases discretized using Lagrangian particles.

I.12 EXTREME

Dataset Description: This dataset contains three-dimensional supernova explosion simulations in a monatomic ideal gas that mocks the interstellar medium of the Milky Way Galaxy. At the beginning of each simulation, the thermal energy of a supernova is deposited at the center of the simulation box; the resulting hot ($\sim 10^7$ K) gas is immediately accelerated and drives a blastwave into the surrounding medium. Because the hot gas becomes supersonic, the physical quantities span roughly seven orders of magnitude.

Modeling Objective: Let $x_t \in \mathbb{R}^{6\times64\times64\times64}$ denote the state of the system at time t on a $64 \times 64 \times 64$ spatial discretization grid. The six feature channels comprise pressure (a scalar field, 1 channel), density (a scalar field, 1 channel), temperature (a scalar field, 1 channel), and velocity (a vector field, 3 channels). Our objective is to develop a neural simulator, defined as a parameterized function Ψ, that maps the current system state to the subsequent state at the next timestep. Formally, we aim to predict $\hat{x}_{t+1} = \Psi(x_t)$, where the predicted output $\hat{x}_{t+1}$ retains the same spatial and feature dimensions $(6 \times 64 \times 64 \times 64)$ as the input state.

Dataset Specific Challenges: The data is fully three-dimensional: each sample contains over 1.5 million values across six channels, and the blastwave propagates and interacts along all spatial axes, making both the memory footprint and the modeling of 3D couplings difficult for standard neural PDE solvers. Moreover, the physical quantities are extremely heavy-tailed and spatially localized, with values that are moderate over most of the domain but explode near the explosion center, resembling a sharply peaked Gaussian profile. For instance, density ranges from 4.7×10^{-14} to 6.7×10^2 (mean 2.9×10^{-1}), pressure from 9.1×10^{-17} to 3.2×10^5 (mean 3.4×10^{-2}), and temperature from 1.0×10^1 to 8.2×10^8 (mean 1.1×10^5), while velocity components span roughly -2.7×10^2 to 2.0×10^2. Such multi-order-of-magnitude dynamic ranges render naive normalization ineffective and destabilize both training and rollout predictions.

Dataset Generation: We adopt the supernova explosion dataset from Ohana et al. (2024). The dataset is generated with the N-body/Smoothed Particle Hydrodynamics (SPH). Because resolving the propagation of the supersonic blastwave demands tiny timesteps and a very large number of integration steps, the Lagrangian SPH particle data are subsequently interpolated onto a uniform Eulerian grid and temporally coarsened for storage. In this work we use the 64^3 version.

Governing Physics: The supernova explosion is governed by the equations of self-gravitating, radiatively-cooled compressible hydrodynamics for a monatomic ideal gas, written in the Lagrangian form solved by the SPH code. The gas obeys the ideal equation of state with specific heat ratio $\gamma = 5/3$,

$$P = (\gamma - 1)\,\rho\,u, \tag{59}$$

where P is the pressure, ρ the smoothed density, and u the specific internal energy. Writing $\frac{d}{dt}$ for the Lagrangian (material) derivative following a fluid element, the conservation of mass, momentum, and energy read

$$\frac{d\rho}{dt} = -\rho\,\nabla\cdot\mathbf{v}, \tag{60}$$

$$\frac{d^2\mathbf{r}}{dt^2} = -\frac{\nabla P}{\rho} + \mathbf{a}_{\text{visc}} - \nabla\Phi, \tag{61}$$

$$\frac{du}{dt} = -\frac{P}{\rho}\,\nabla\cdot\mathbf{v} + \frac{\Gamma - \Lambda}{\rho}, \tag{62}$$

where $\mathbf{r}$ is the position of a fluid element and $\mathbf{v} = d\mathbf{r}/dt$ its velocity. In the momentum equation, $-\nabla P/\rho$ is the pressure-gradient acceleration, $\mathbf{a}_{\text{visc}}$ is the acceleration generated by the artificial viscosity used to capture shocks, and $-\nabla\Phi$ is the acceleration due to self-gravity, with Φ the gravitational potential sourced by the gas distribution. In the energy equation, the first term is the reversible PdV work done under compression or expansion, while $(\Gamma - \Lambda)/\rho$ is the net radiative source term: Γ is the radiative heat influx per unit volume and Λ the radiative heat outflux (cooling) per unit volume. It is this radiative sink Λ that rapidly cools the shocked, compressed shells behind the blastwave, and the stiff, highly localized nature of $\Gamma - \Lambda$ together with the supersonic velocities is what forces the very small timesteps required to integrate the system.

Experiment Results: Table 27 details the full experimental results, highlighting the superior performance of the proposed design. We exclude GNOT from this comparison because its training

diverges due to the extreme values present in the dataset. Furthermore, AMG is omitted due to computational constraints. Its graph-based architecture scales poorly to large 3D datasets, leading to infeasible training times.

	RMSE	FRMSE	Kinetic Energy
GNOT	-	-	-
Transolver	2.289×10^5	1.090×10^8	1.736×10^1
Transolver++	2.190×10^5	1.044×10^8	1.053×10^2
LNO	4.813×10^5	2.315×10^8	2.293×10^5
AMG	-	-	-
Agent	$\mathbf{1.340 \times 10^5}$	$\mathbf{5.065 \times 10^7}$	$\mathbf{5.996 \times 10^{-1}}$

Table 27: Performance comparison on the **Extreme** dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 18 for visualizations of the dataset.

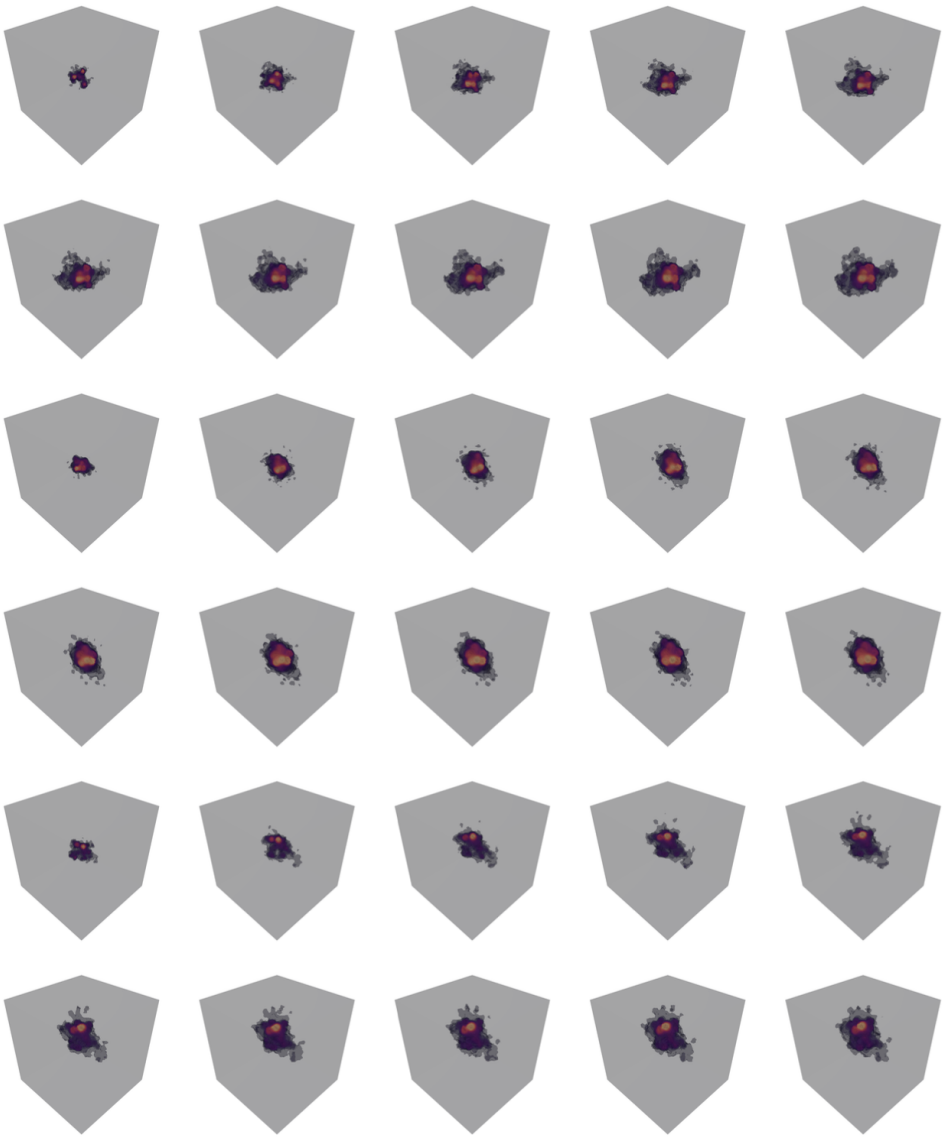


Figure 18: Visualization of the **Extreme** dataset. It displays a supernova explosion.

I.13 MIXEDRES

Dataset Description: This dataset contains two-dimensional incompressible Navier–Stokes dynamics in vorticity form on periodic rectangular domains of varying size and resolution. Each trajectory is simulated at Reynolds number 500 under a fixed sinusoidal forcing, with the domain side lengths drawn independently from $\{\pi, 2\pi, 3\pi, 4\pi\}$ and the output resolution along each axis drawn independently from $\{32, 64, 128, 256, 512\}$. All trajectories are computed on a high-resolution 1024×1024 grid and subsampled.

Modeling Objective: Let $x_t \in \mathbb{R}^{1\times H\times W}$ denote the vorticity field at time t, where the spatial dimensions H and W vary across trajectories. The model additionally receives a positional field $p \in \mathbb{R}^{2\times H\times W}$ containing the physical coordinates of each grid point. Our objective is to develop a neural simulator, defined as a parameterized function Ψ, that maps the current vorticity field and its positional coordinates to the subsequent state at the next timestep. Formally, we aim to predict $\hat{x}_{t+1} = \Psi(x_t, p)$, where the predicted output $\hat{x}_{t+1} \in \mathbb{R}^{1\times H\times W}$ retains the same spatial and feature dimensions as the input state.

Dataset Specific Challenges: The same discrete array can represent physically different fields depending on the domain: a 128×128 grid may span a domain of length π or 4π, changing the physical wavelengths, effective grid spacing, and the amount of advection per time step. Consequently, the model cannot treat pixel indices as physical positions and must instead infer the metric of the grid from the positional input, effectively encoding dx and dy into its learned dynamics. Moreover, the number of grid points per sample spans a $256\times$ range, from 32×32 (1,024 points) to 512×512 (262,144 points), so the computational and memory footprint of a forward pass varies drastically across samples. The model and training pipeline must therefore remain efficient at both extremes: small samples should not be dominated by fixed overhead, while the largest samples must fit within memory.

Dataset Generation: The dataset generation follows Section I.5, where we solve the 2D incompressible Navier-Stokes equations in vorticity form using a pseudo-spectral method. We randomly sample the spatial domain size and resolution from $\{\pi, 2\pi, 3\pi, 4\pi\}$ and $\{32, 64, 128, 256, 512\}$, respectively. To ensure the simulation is well-resolved, we compute all trajectories at a 1024×1024 resolution and uniformly downsample them to the target resolution.

Governing Physics: The governing physics follow the formulation presented in Section I.5.

Experiment Results: We present the results in Table 28. Because the data is Eulerian, we additionally report the FRMSE and Kinetic Energy. The results show that our agent consistently achieves superior performance. AMG is not applicable because our dataset includes high-resolution 512×512 data, causing its training to take 28,000 seconds per epoch, which is computationally infeasible.

	RMSE	FRMSE	Kinetic Energy
GNOT	3.629×10^{0}	8.521×10^{2}	1.508×10^{0}
Transolver	3.616×10^{0}	8.217×10^{2}	1.415×10^{0}
Transolver++	3.564×10^{0}	7.573×10^{2}	1.166×10^{0}
LNO	3.481×10^{0}	7.548×10^{2}	1.125×10^{0}
AMG	-	-	-
Agent	$\mathbf{3.455 \times 10^{0}}$	$\mathbf{7.014 \times 10^{2}}$	$\mathbf{6.830 \times 10^{-1}}$

Table 28: Performance comparison on the **MixedRes** dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 19 for visualizations of the dataset.

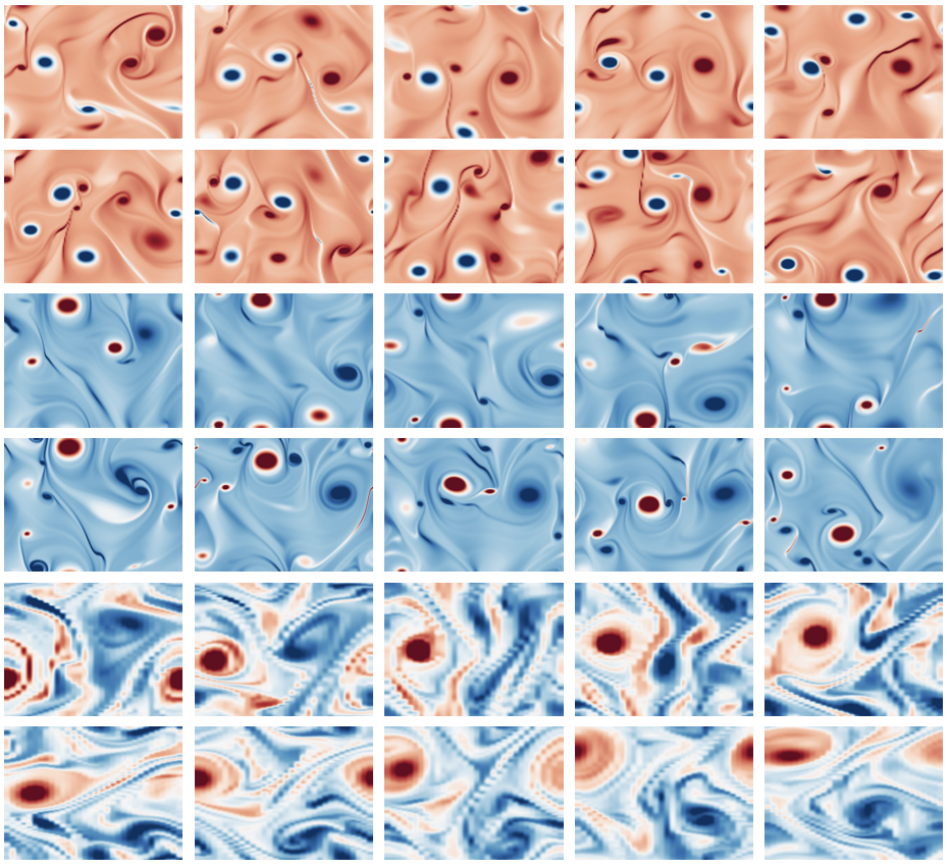


Figure 19: Visualization of the **MixedRes** dataset. The figure displays Navier-Stokes dataset in vorticity form at different spatial sizes and resolutions.

I.14 ExtLag1

Dataset Description: This dataset contains liquid sloshing dynamics in Lagrangian particle format. Fluid particles are enclosed in a sealed tank that undergoes external excitation, causing the free surface to slosh. The tank motion is prescribed as a time series of rigid-body accelerations, and the fluid responds with large free-surface deformation, wave breaking, and impacts against the tank walls. The sloshing motion is provided with the linear accelerations and the angular accelerations.

Modeling Objective: Let $x_t \in \mathbb{R}^{N\times 54}$ denote the state of the system at time t, where N is the number of fluid particles. The 54 features per particle comprise the last-step position (2), the historical velocity over 5 frames ($5 \times 2 = 10$), and the sloshing motion over 6 historical frames together with the next-step sloshing motion ($7 \times 6 = 42$). Our objective is to develop a neural simulator, defined as a parameterized function Ψ, that maps the current system state to the particle velocity at the next timestep. Formally, we aim to predict $\hat{v}_{t+1} = \Psi(x_t)$, where the predicted output $\hat{v}_{t+1} \in \mathbb{R}^{N\times 2}$ is the particle velocity at the next time step.

Dataset Specific Challenges: The sloshing motion is a global, per-frame signal shared by all particles rather than a per-particle quantity. The standard approach simply concatenates it onto every particle's node features, which treats a global driving force as local information and dilutes it among high-dimensional particle states. Addressing this problem therefore requires innovative conditioning mechanisms that inject the excitation signal into the neural solver more effectively.

Dataset Generation: We generate the sloshing dataset using weakly-compressible Smoothed Particle Hydrodynamics (SPH) solver (Zhan et al., 2025; Domínguez et al., 2022). The simulation is two-dimensional with an inter-particle spacing of $dp = 3 \times 10^{-3}\,\mathrm{m}$, yielding a smoothing length $h = 0.91924\sqrt{2\,dp^2}$, and a total of 9908 particles (938 fixed boundary particles and 8970 fluid particles). Time integration uses a Symplectic scheme together with the Wendland kernel, and the artificial-viscosity treatment is applied with coefficient $\alpha = 0.05$. The weakly-compressible pressure closure uses a polytropic exponent $\gamma = 7$ with reference density $\rho_0 = 1000\,\mathrm{kg/m^3}$, and the numerical speed of sound is set to 30 times the characteristic system speed. Density evolution is stabilized with the full Fourtakas density-diffusion term (coefficient 0.1), while particle shifting is disabled. The timestep is adaptive under a CFL coefficient of 0.2 with $\mathrm{dtmin} = 0.05\,h/c_s$, and particles are rejected outside the density bounds $[700, 1300]\,\mathrm{kg/m^3}$. The external sloshing excitation is imposed on the fluid through a prescribed time-dependent rigid acceleration read from a driving file, applied about the tank reference centre with global gravity retained.

Governing Physics: The sloshing case is governed by the weakly-compressible Navier–Stokes equations for a single liquid phase, solved in the Lagrangian SPH frame. Writing $\frac{d}{dt}$ for the material derivative following a fluid element, the conservation of mass and momentum read

$$\frac{d\rho}{dt} = -\rho\,\nabla\cdot\mathbf{v}, \tag{63}$$

$$\frac{d\mathbf{v}}{dt} = -\frac{1}{\rho}\nabla P + \nu\nabla^2\mathbf{v} + \mathbf{g} + \mathbf{a}_{\mathrm{ext}}(t), \tag{64}$$

where ρ is the density, $\mathbf{v}$ the velocity, P the pressure, ν the kinematic viscosity, and $\mathbf{g} = (0, 0, -9.81)^\top\,\mathrm{m/s^2}$ the gravitational acceleration. The final term $\mathbf{a}_{\mathrm{ext}}(t)$ is the externally imposed sloshing forcing: the fluid is driven not by moving the domain boundaries directly but by adding a prescribed, time-dependent body acceleration to every fluid element. Expressed about the tank reference centre $\mathbf{r}_c$, this excitation combines a translational and a rotational contribution,

$$\mathbf{a}_{\mathrm{ext}}(t) = \mathbf{a}_T(t) + \boldsymbol{\alpha}(t)\times(\mathbf{r}-\mathbf{r}_c) + \boldsymbol{\omega}(t)\times\big(\boldsymbol{\omega}(t)\times(\mathbf{r}-\mathbf{r}_c)\big) + 2\,\boldsymbol{\omega}(t)\times\mathbf{v}, \tag{65}$$

where $\mathbf{a}_T(t)$ is the imposed linear acceleration of the tank, $\boldsymbol{\omega}(t)$ and $\boldsymbol{\alpha}(t) = d\boldsymbol{\omega}/dt$ are the imposed angular velocity and angular acceleration, and the successive terms represent the translational, Euler, centrifugal, and Coriolis accelerations induced by the moving reference frame. The time histories of $\mathbf{a}_T(t)$ and $\boldsymbol{\omega}(t)$ are supplied externally from the acceleration driving file, so that $\mathbf{a}_{\mathrm{ext}}(t)$ acts as the sole time-dependent driver that continually sloshes the liquid against the tank walls. The system is closed by the weakly-compressible equation of state relating pressure to density,

$$P = B\left[\left(\frac{\rho}{\rho_0}\right)^{\gamma} - 1\right], \qquad B = \frac{c_s^2\,\rho_0}{\gamma}, \tag{66}$$

with reference density ρ_0, polytropic exponent $\gamma = 7$, and numerical speed of sound c_s; the pressure constant evaluates to $B = 1.0973 \times 10^5$ Pa for this configuration.

Experiment Results: The comprehensive experimental results are detailed in Table 29. Given the nature of the Lagrangian particle data, we include LagKE to evaluate physical consistency, as well as the Sinkhorn metric to quantify the optimal transport distance between the predicted and ground-truth particle distributions. Furthermore, we report Eulerian aggregated metrics (EuARMSE and EuAFRMSE) to assess the practical utility of the predictions. The agent consistently achieves strong results across all evaluated metrics, highlighting the robustness and superiority of its design.

	RMSE	LagKE	Sinkhorn	EuARMSE	EuAFRMSE
GNOT	2.176×10^{-2}	1.152×10^{1}	3.882×10^{-4}	4.715×10^{-3}	1.431×10^{-1}
Transolver	3.526×10^{-2}	3.468×10^{1}	1.144×10^{-3}	6.883×10^{-3}	2.143×10^{-1}
Transolver++	1.729×10^{-2}	6.427×10^{0}	2.568×10^{-4}	3.571×10^{-3}	1.071×10^{-1}
LNO	1.407×10^{-2}	3.814×10^{0}	1.868×10^{-4}	2.730×10^{-3}	8.317×10^{-2}
AMG	1.892×10^{-2}	5.801×10^{0}	3.067×10^{-4}	3.941×10^{-3}	1.152×10^{-1}
Agent	$\mathbf{1.954 \times 10^{-3}}$	$\mathbf{4.650 \times 10^{-2}}$	$\mathbf{4.286 \times 10^{-6}}$	$\mathbf{4.394 \times 10^{-4}}$	$\mathbf{1.005 \times 10^{-2}}$

Table 29: Performance comparison on the **ExtLag1** dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 20 for visualizations of the dataset.

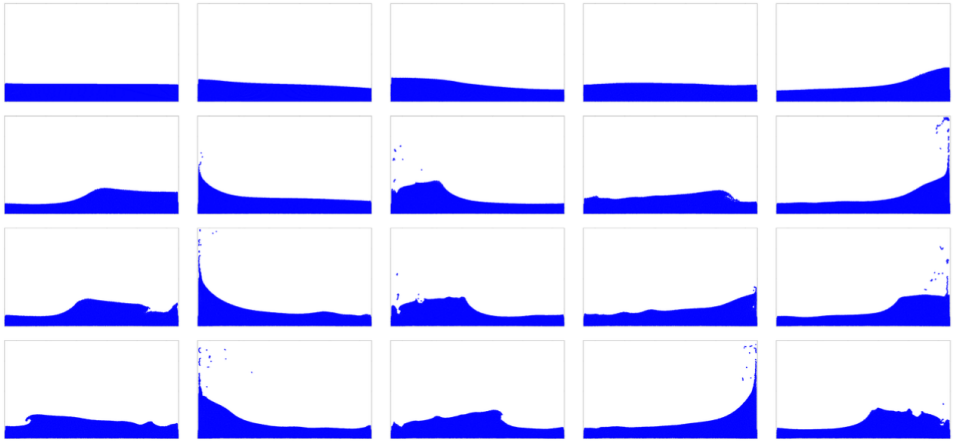


Figure 20: Visualization of the **ExtLag1** dataset. The dataset features Lagrangian fluid particles contained within a closed box subject to external sloshing motion.

I.15 EXTLAG2

Dataset Description: This dataset contains piston-driven fluid dynamics in Lagrangian particle format. A piston at the left boundary of the tank undergoes prescribed horizontal motion, pushing the fluid and driving waves through the domain. Each particle is labeled by type (0 = fluid, 1 = piston): fluid particles obey the free dynamics, while piston particles follow a known rigid trajectory that acts as the external forcing of the system.

Modeling Objective: Let $x_t \in \mathbb{R}^{N\times 15}$ denote the state of the system at time t, where N is the number of particles. The 15 features per particle comprise the last-step position (2), the historical velocity over 5 frames ($5 \times 2 = 10$), the next-step velocity (2), and a piston indicator column (1). The next-step velocity slot is a known-driver feature: for piston particles it contains the true $t+1$ velocity of the prescribed motion, while for fluid particles it is zero since $t+1$ is exactly the quantity to be predicted; the indicator column lets the model distinguish a masked zero from a genuinely zero driver. Our objective is to develop a neural simulator, defined as a parameterized function Ψ, that maps the current system state to the particle velocity at the next timestep. Formally, we aim to predict $\hat{v}_{t+1} = \Psi(x_t)$, where the predicted output $\hat{v}_{t+1} \in \mathbb{R}^{N\times 2}$ is the particle velocity at the next time step. The loss is evaluated on fluid particles only, since the piston motion is prescribed rather than predicted.

Dataset Specific Challenges: The fluid is driven entirely by the piston boundary: momentum enters the system through a thin layer of fluid particles in contact with the piston and must propagate correctly through the bulk. The model must accurately handle this external force.

Dataset Generation: We generate dataset with a weakly-compressible Smoothed Particle Hydrodynamics (SPH) solver (Zhan et al., 2025; Domínguez et al., 2022). The simulation is two-dimensional with an inter-particle spacing of $dp = 3 \times 10^{-2}\,\mathrm{m}$, yielding a smoothing length $h = 1.2\sqrt{2\,dp^2}$, and a total of 6865 particles (385 fixed boundary particles, 201 moving boundary particles forming the piston, and 6279 fluid particles). Time integration uses a Symplectic scheme together with the Wendland kernel, and the artificial-viscosity treatment is applied with coefficient $\alpha = 0.01$. The weakly-compressible pressure closure uses a polytropic exponent $\gamma = 7$ with reference density $\rho_0 = 1000\,\mathrm{kg/m^3}$, and the numerical speed of sound is set to 20 times the characteristic system speed. Density evolution is stabilized with the Fourtakas density-diffusion term (coefficient 0.1), while particle shifting is disabled. The timestep is adaptive under a CFL coefficient of 0.2 with $\mathrm{dtmin} = 0.05\,h/c_s$, and particles are rejected outside the density bounds $[700, 1300]\,\mathrm{kg/m^3}$. The waves are driven by a piston wavemaker: the moving boundary is actuated by a second-order regular-wave generation routine configured with wave height $H = 0.15\,\mathrm{m}$, wave period $T = 2\,\mathrm{s}$, and still-water depth $d = 0.66\,\mathrm{m}$, ramped in over one wave period, with the piston displacing along the x-direction.

Governing Physics: The data is governed by the weakly-compressible Navier–Stokes equations for a single liquid phase, solved in the Lagrangian SPH frame. Writing $\frac{d}{dt}$ for the material derivative following a fluid element, the conservation of mass and momentum read

$$\frac{d\rho}{dt} = -\rho\,\nabla\cdot\mathbf{v}, \tag{67}$$

$$\frac{d\mathbf{v}}{dt} = -\frac{1}{\rho}\nabla P + \nu\nabla^2\mathbf{v} + \mathbf{g}, \tag{68}$$

where ρ is the density, $\mathbf{v}$ the velocity, P the pressure, ν the kinematic viscosity, and $\mathbf{g} = (0, 0, -9.81)^\top\,\mathrm{m/s^2}$ the gravitational acceleration. The system is closed by the weakly-compressible equation of state relating pressure to density,

$$P = B\left[\left(\frac{\rho}{\rho_0}\right)^{\gamma} - 1\right], \qquad B = \frac{c_s^2\,\rho_0}{\gamma}, \tag{69}$$

with reference density ρ_0, polytropic exponent $\gamma = 7$, and numerical speed of sound c_s; the pressure constant evaluates to $B = 3.3634 \times 10^5\,\mathrm{Pa}$ for this configuration.

Unlike the interior fluid, the waves are not forced through a body term but through a moving solid boundary: a subset of boundary particles forms a piston whose position $X_p(t)$ is prescribed in time and imposed as a Dirichlet velocity condition on the fluid. For second-order regular-wave generation, the piston displacement combines the first-order Biesel stroke with a second-order Stokes

correction,

$$X_p(t) = \frac{S}{2}\sin(\omega t + \phi) + \frac{S^{(2)}}{4}\sin(2\omega t + \phi), \qquad \dot{X}_p(t) = \mathbf{v}\cdot\mathbf{e}_x\big|_{\text{piston}}, \tag{70}$$

where $\omega = 2\pi/T$ is the angular frequency of the target wave of period $T = 2\,\mathrm{s}$ and height $H = 0.15\,\mathrm{m}$, ϕ is the initial phase, and $\mathbf{e}_x$ is the direction of piston travel. The first-order stroke S follows the Biesel transfer function for a piston-type wavemaker at still-water depth $d = 0.66\,\mathrm{m}$,

$$\frac{H}{S} = \frac{2\big(\cosh(2kd) - 1\big)}{\sinh(2kd) + 2kd}, \tag{71}$$

with the wavenumber k fixed by the linear dispersion relation $\omega^2 = g\,k\,\tanh(kd)$, and $S^{(2)}$ is the corresponding second-order stroke amplitude that suppresses spurious free harmonics. A ramp function multiplies $X_p(t)$ over the first wave period so that the excitation is applied smoothly from rest. The prescribed piston motion is thus the sole time-dependent driver that generates the propagating wave train down the flume.

Experiment Results: We present the complete experimental results in Table 30. As with other Lagrangian particle datasets, we additionally report LagKE, Sinkhorn, EuARMSE, and EuAFRMSE. The results show that the agent's design achieves superior performance across all of these metrics.

	RMSE	LagKE	Sinkhorn	EuARMSE	EuAFRMSE
GNOT	1.365×10^{-3}	5.047×10^{-2}	6.958×10^{-4}	2.844×10^{-4}	6.658×10^{-3}
Transolver	1.456×10^{-3}	4.405×10^{-2}	6.952×10^{-4}	2.978×10^{-4}	7.065×10^{-3}
Transolver++	1.551×10^{-3}	7.764×10^{-2}	6.963×10^{-4}	3.396×10^{-4}	8.152×10^{-3}
LNO	1.640×10^{-3}	4.649×10^{-2}	6.968×10^{-4}	3.008×10^{-4}	6.991×10^{-3}
AMG	1.566×10^{-3}	4.806×10^{-2}	6.964×10^{-4}	2.748×10^{-4}	6.766×10^{-3}
Agent	$\mathbf{5.483 \times 10^{-4}}$	$\mathbf{1.712 \times 10^{-2}}$	$\mathbf{6.427 \times 10^{-4}}$	$\mathbf{9.468 \times 10^{-5}}$	$\mathbf{2.266 \times 10^{-3}}$

Table 30: Performance comparison on the **ExtLag2** dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 21 for visualizations of the dataset.

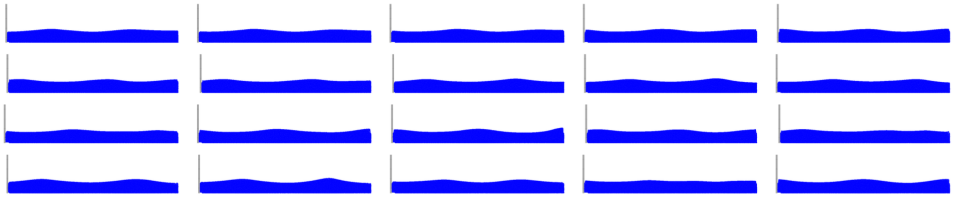


Figure 21: Visualization of the **ExtLag2** dataset. The dataset features a horizontally moving piston on the right side that drives the water.

I.16 MIXEDDIM

Dataset Description: This dataset contains solutions of the Allen–Cahn equation, a reaction–diffusion PDE describing phase separation, in one, two, and three spatial dimensions. The underlying physics is identical across dimensions: a diffusion term competes with a nonlinear double-well reaction term, driving the field toward $+1/-1$ phases separated by thin diffuse interfaces. Each trajectory is discretized on a regular grid with 64 points per spatial axis, giving 64 nodes in 1D, $64^2 = 4096$ nodes in 2D, and $64^3 = 262144$ nodes in 3D.

Modeling Objective: Let $x_t \in \mathbb{R}^{N\times 1}$ denote the system state at time t, where N is the number of grid nodes ($N = 64$, 64^2, or 64^3 depending on the dimension) and the single feature channel is the scalar phase field u. The data is stored flattened with shape $(\text{num_traj}, T, N)$ and can be reshaped back to its grid form. Our objective is to learn a single parameterized neural solver Ψ such that $\hat{x}_{t+1} = \Psi(x_t)$, where the predicted output $\hat{x}_{t+1}$ retains the same shape $(N, 1)$ as the input state. Critically, one model with a single set of weights must generalize across all three dimensions, exploiting the fact that the governing equation is shared.

Dataset Specific Challenges: The solver must be dimension-agnostic: it cannot assume a fixed grid shape, a fixed input size, or a dimension-specific convolutional structure, and must instead learn a representation of the shared Allen–Cahn dynamics that transfers across 1D, 2D, and 3D discretizations. Compounding this, the number of nodes spans over three orders of magnitude (from 64 to 262144), so the model must handle both very small and very large inputs. Memory management is therefore critical: attention or global coupling mechanisms that scale quadratically in the number of nodes may be infeasible in 3D, while overly local models may underfit the 1D case.

Dataset Generation: We generate the Allen–Cahn dataset via Exponential Time Differencing Runge–Kutta (ETDRK) schemes. The spatial derivatives are treated spectrally: the linear diffusion operator is diagonalized in Fourier space and integrated exactly, while the nonlinear double-well reaction term is evaluated pseudo-spectrally, yielding a stiffly-accurate, high-order-in-time solver that is identical in structure across all spatial dimensions. For each of the 1D, 2D, and 3D cases we simulate on a periodic domain $[0, 1)^d$ discretized with 64 points per axis and advance with a fixed timestep $\Delta t = 0.1$. We use a diffusivity of $\nu = 5 \times 10^{-3}$, a linear reaction coefficient $c_1 = 1.0$, and a cubic reaction coefficient $c_3 = -1.0$. Initial conditions are drawn from a random truncated Fourier series with wavenumber cutoff 5, producing smooth random fields that subsequently coarsen into $+1/-1$ phase domains. In every dimension we roll out 30 trajectories of 100 frames each (excluding the initial state), giving 3000 frames per dimension.

Governing Physics: The Allen–Cahn equation is a reaction–diffusion PDE modeling phase separation, in which a linear diffusion term competes with a nonlinear double-well reaction term to drive the scalar phase field $u(\mathbf{x}, t)$ toward the two stable phases $u = +1$ and $u = -1$, separated by thin diffuse interfaces. On the periodic domain $\Omega = [0, 1)^d$ it is written as

$$\frac{\partial u}{\partial t} = \nu\, \Delta u + c_1\, u + c_3\, u^3, \qquad \mathbf{x} \in \Omega,\ t > 0, \tag{72}$$

where $\Delta = \sum_{i=1}^{d} \partial^2/\partial x_i^2$ is the d-dimensional Laplacian and $\nu > 0$ is the diffusivity controlling the width of the interfaces. The reaction drives the field toward these two wells while the diffusion penalizes sharp gradients, and the balance between the two sets the characteristic interface thickness $\sim \sqrt{\nu}$. The governing equation is identical in form across all three spatial dimensions, differing only in the dimensionality of the Laplacian Δ; this shared structure is precisely what a single dimension-agnostic solver is meant to exploit.

Experiment Results: We present the complete experimental results in Table 31. AMG is not applicable to this dataset because its graph-based architecture scales poorly to 3D data, resulting in impractical training times. Overall, the proposed agent design achieves superior performance across all dimensions.

Visualization: We refer the readers to Figure 22 for visualizations of the dataset.

	RMSE	FRMSE	RMSE (1D)	RMSE (2D)	RMSE (3D)
GNOT	3.615×10^{-2}	3.521×10^{0}	6.239×10^{-2}	2.752×10^{-2}	1.855×10^{-2}
Transolver	3.381×10^{-2}	3.470×10^{0}	5.688×10^{-2}	2.732×10^{-2}	1.722×10^{-2}
Transolver++	3.357×10^{-2}	3.766×10^{0}	5.186×10^{-2}	3.031×10^{-2}	1.853×10^{-2}
LNO	5.048×10^{-2}	3.937×10^{0}	1.014×10^{-1}	3.131×10^{-2}	1.870×10^{-2}
AMG	-	-	-	-	-
Agent	$\mathbf{9.418 \times 10^{-3}}$	$\mathbf{6.707 \times 10^{-1}}$	$\mathbf{1.753 \times 10^{-2}}$	$\mathbf{7.679 \times 10^{-3}}$	$\mathbf{3.049 \times 10^{-3}}$

Table 31: Performance comparison on the **MixedDim** dataset. Best results are shown in **bold**.

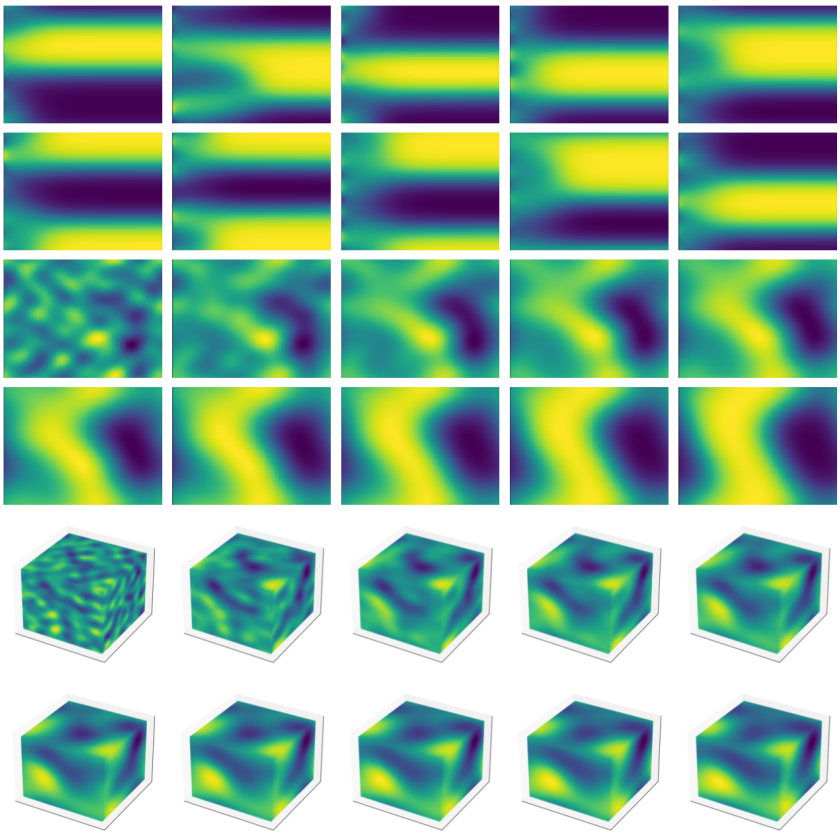


Figure 22: Visualization of the **MixedDim** dataset. This figure displays the Allen-Cahn PDE across three spatial dimensions: 1D (first two rows), 2D (middle two rows), and 3D (final two rows). All datasets share the same governing physics and differ only in dimensionality. For visualization purposes in the 1D case, the x-axis corresponds to time and the y-axis corresponds to the spatial discretization grid.

I.17 MIXEDBC

Dataset Description: This dataset contains two-dimensional linear advection dynamics evolving on a 128×128 spatial grid under heterogeneous boundary conditions. Each trajectory is governed by one of three boundary condition families—Dirichlet, Neumann, or periodic—with time-dependent boundary data in the non-periodic cases. Unlike standard benchmarks where a single boundary treatment is fixed across the entire dataset, trajectories here differ in both the type of boundary condition and the values imposed on each of the four domain edges, reflecting practical settings where the same interior physics must be simulated under diverse and externally driven boundary constraints.

Modeling Objective: Let $x_t \in \mathbb{R}^{1\times128\times128}$ denote the system state at time t. The model additionally receives a boundary type identifier $b \in \{0, 1, 2\}$, encoding Dirichlet, Neumann, or periodic conditions respectively, and a boundary value vector $v_t \in \mathbb{R}^{1\times4}$ specifying the boundary data on the (left, right, bottom, top) edges at time t: prescribed state values for Dirichlet, prescribed fluxes for Neumann, and zero-padding for periodic. Our objective is to learn a parameterized neural simulator Ψ such that $\hat{x}_{t+1} = \Psi(x_t, b, v_t)$, where the predicted output $\hat{x}_{t+1} \in \mathbb{R}^{1\times128\times128}$ is the state at the next time step, matching the shape of the input state.

Dataset Specific Challenges: The three boundary condition families induce qualitatively different dynamics near the domain edges: Dirichlet conditions exactly define the state, Neumann conditions constrain its normal gradient, and periodic conditions couple opposite edges through wrap-around transport. A single model must internalize all three behaviors and switch between them based on a discrete conditioning input, rather than specializing to one edge treatment. Compounding this, the boundary value vector carries a type-dependent semantic: the same numerical entries represent state values under Dirichlet conditions, fluxes under Neumann conditions, and uninformative padding under periodic conditions. The model must therefore learn to interpret this shared input slot conditionally, using the boundary type to modulate how the boundary values influence the predicted interior dynamics.

Dataset Generation: We generate this dataset for the two-dimensional linear advection equation, producing 30 trajectories of 100 frames each for every boundary condition family on a 128×128 grid over the unit square $[0, 1]^2$ with a frame interval of $\Delta t = 0.1$ (total time $T = 10$). A single divergence-free-scaled velocity field is drawn once from a Gaussian random field and reused across all trajectories and all three boundary families, so that the interior transport is held fixed and only the boundary treatment varies; its amplitude is rescaled to a target Courant number of 0.45. For each trajectory the initial phase field is sampled independently from the same Gaussian random field prior. The non-periodic families additionally receive, on each of the four edges, an independent time-dependent boundary signal built as a low-frequency truncated Fourier series, clipped to a physical range; these per-edge signals are recorded frame-by-frame in the order (left, right, bottom, top). The Dirichlet and Neumann cases are integrated with an explicit finite-volume upwind scheme under an adaptive CFL-limited sub-stepping loop between saved frames, whereas the periodic case is integrated with a pseudo-spectral Fourier method using classical fourth-order Runge–Kutta time stepping and $2/3$-rule dealiasing. The three families are stored together with a boundary type identifier $b \in \{0, 1, 2\}$ (Dirichlet, Neumann, periodic) and, for the non-periodic cases, the recorded boundary value vectors; periodic trajectories carry no boundary data and are zero-padded in that slot.

Governing Physics: The dataset is governed by the two-dimensional linear advection (transport) equation for a scalar field $u(\mathbf{x}, t)$ carried by a prescribed, time-independent velocity field $\mathbf{a}(\mathbf{x}) = (a_x(\mathbf{x}), a_y(\mathbf{x}))$,

$$\frac{\partial u}{\partial t} + \nabla \cdot (\mathbf{a}\, u) = 0, \qquad \mathbf{x} \in \Omega = [0, 1]^2,\ t > 0, \tag{73}$$

which, since the velocity field used here is constructed to be (discretely) divergence-free, is equivalent to the non-conservative form $\partial_t u + \mathbf{a} \cdot \nabla u = 0$. The equation simply transports the initial field along the characteristics $\mathrm{d}\mathbf{x}/\mathrm{d}t = \mathbf{a}(\mathbf{x})$, so that u is constant along each streamline and the interior dynamics are governed entirely by the fixed advecting velocity. What distinguishes the trajectories

is the closure imposed on the domain boundary $\partial\Omega$, drawn from three families:

$$\underbrace{u = g(\mathbf{x}, t)}_{\text{Dirichlet}}, \qquad \underbrace{\frac{\partial u}{\partial n} = h(\mathbf{x}, t)}_{\text{Neumann}}, \qquad \underbrace{u(\mathbf{x} + \mathbf{L}) = u(\mathbf{x})}_{\text{periodic}}, \tag{74}$$

where n denotes the outward normal, $\mathbf{L}$ the domain periods, and g and h are the prescribed time-dependent boundary data supplied edge-by-edge on the (left, right, bottom, top) walls. The Dirichlet condition pins the boundary state to g; the Neumann condition prescribes the normal derivative $\partial u/\partial n = h$, imposed numerically through one-sided ghost values; and the periodic condition couples opposite edges through wrap-around transport, requiring no external boundary data. Because the interior operator is identical across all trajectories, the differences observed near the domain edges are attributable solely to the boundary condition type and its associated data.

Experiment Results: We present the complete experimental results in Table 32. Because this dataset relies on an Eulerian grid, we also include the FRMSE and mass error. To isolate the effects of the boundary conditions, we report separate RMSE values for the Dirichlet, Neumann, and periodic conditions, alongside the RMSE evaluated exclusively at the boundary points. Overall, the proposed agent design achieves superior performance across all metrics.

	RMSE	FRMSE	Mass Error	RMSE (Dirichlet)
GNOT	1.383×10^{-1}	1.556×10^{1}	4.417×10^{2}	1.464×10^{-1}
Transolver	8.500×10^{-2}	9.776×10^{0}	2.058×10^{2}	1.417×10^{-1}
Transolver++	8.498×10^{-2}	9.771×10^{0}	2.025×10^{2}	1.418×10^{-1}
LNO	8.511×10^{-2}	9.786×10^{0}	2.043×10^{2}	1.421×10^{-1}
AMG	2.064×10^{-1}	2.290×10^{1}	8.483×10^{2}	1.348×10^{-1}
Agent	$\mathbf{4.435 \times 10^{-2}}$	$\mathbf{4.579 \times 10^{0}}$	$\mathbf{6.910 \times 10^{1}}$	$\mathbf{6.809 \times 10^{-2}}$

	RMSE (Neumann)	RMSE (Periodic)	RMSE (BC)
GNOT	5.262×10^{-2}	1.822×10^{-1}	7.049×10^{-1}
Transolver	1.550×10^{-2}	3.661×10^{-2}	4.461×10^{-1}
Transolver++	1.505×10^{-2}	3.640×10^{-2}	4.461×10^{-1}
LNO	1.503×10^{-2}	3.638×10^{-2}	4.467×10^{-1}
AMG	1.057×10^{-1}	3.138×10^{-1}	9.335×10^{-1}
Agent	$\mathbf{1.039 \times 10^{-2}}$	$\mathbf{3.403 \times 10^{-2}}$	$\mathbf{2.035 \times 10^{-1}}$

Table 32: Performance comparison on the `MixedBC` dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 23 for visualizations of the dataset.

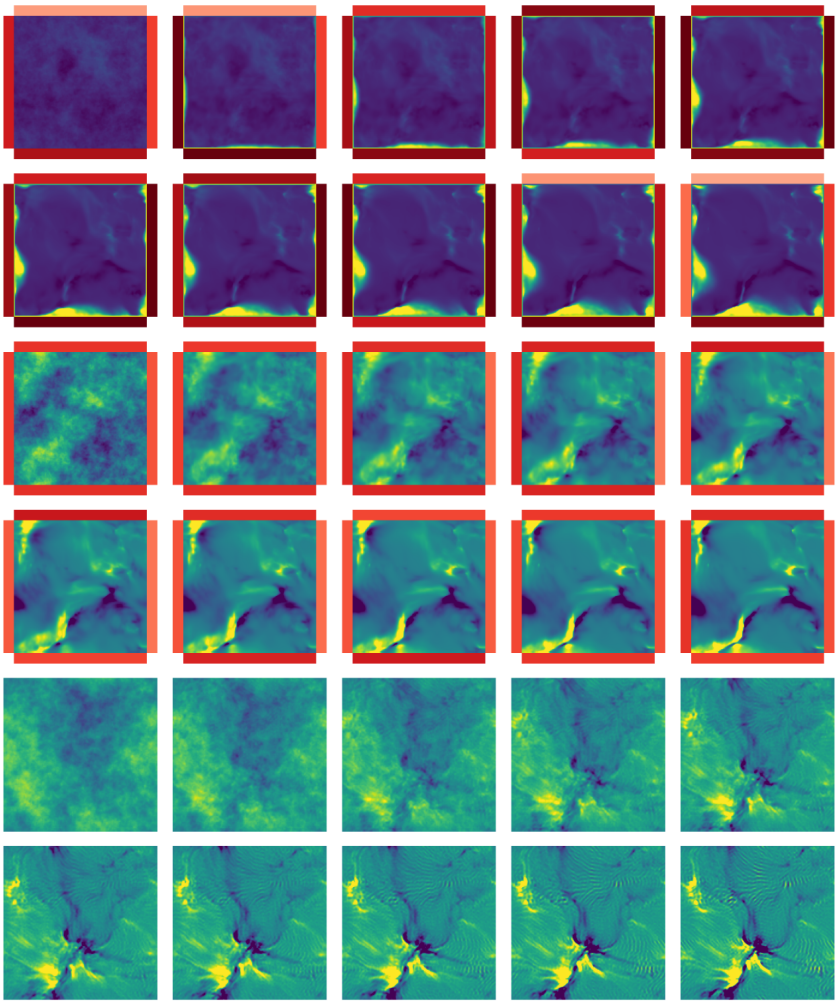


Figure 23: Visualization of the **MixedBC** dataset. The first two rows display data for the advection equation with time-dependent Dirichlet boundary conditions. The middle and final two rows illustrate time-dependent Neumann and periodic boundary conditions, respectively. The underlying PDE is identical across all datasets. They differ solely in their boundary conditions.

I.18 LongHorizon

Dataset Description: This dataset contains solutions of the one-dimensional Kuramoto–Sivashinsky equation, a canonical model of spatiotemporal chaos arising in flame-front propagation and thin-film instabilities. The equation combines a destabilizing second-order diffusion term, a stabilizing fourth-order hyperviscosity term, and a nonlinear advection term; their interplay produces sustained chaotic dynamics with a continuum of interacting cellular structures. The field is discretized on a uniform periodic grid of 256 points, and each trajectory spans 5000 time steps.

Modeling Objective: Let $x_t \in \mathbb{R}^{256\times 1}$ denote the system state at time t, where 256 is the number of spatial grid points and the single channel dimension is the scalar field u. Our objective is to learn a parameterized neural simulator Ψ such that $\hat{x}_{t+1} = \Psi(x_t)$, where the predicted output $\hat{x}_{t+1}$ retains the same shape $(256, 1)$ as the input state. At evaluation time, the model is applied autoregressively for the full 5000 steps, with each prediction fed back as the next input.

Dataset Specific Challenges: Long autoregressive rollouts amplify any small bias or high-frequency artifact in the one-step map. Over 5000 steps, even tiny inconsistencies in the balance between the destabilizing and hyperviscous terms can accumulate into unbounded growth or artificial damping, so temporal stability is as critical as one-step accuracy.

Dataset Generation: We generate the Kuramoto–Sivashinsky dataset using Exponential Time Differencing Runge–Kutta (ETDRK) schemes on periodic domains. The method treats the linear operator (the second-order and fourth-order terms) spectrally and exactly in Fourier space, while the nonlinear advection term is evaluated pseudo-spectrally, yielding a stiffly-accurate, high-order-in-time integrator well suited to the stiffness introduced by the fourth-order hyperviscosity. We simulate on a periodic domain discretized with 256 uniformly spaced points and advance with a fixed time step $\Delta t = 0.5$. Initial conditions are drawn from a random truncated Fourier series with wavenumber cutoff 5, producing smooth random fields that the dynamics subsequently drive into the chaotic regime. We roll out 120 trajectories of 5000 steps each.

Governing Physics: The Kuramoto–Sivashinsky equation is a fourth-order nonlinear PDE modeling spatiotemporal chaos in settings such as laminar flame-front propagation and the dynamics of thin liquid films. On the periodic domain $\Omega = [0, L)$ it is written, in the gradient-norm form integrated here, as

$$\frac{\partial u}{\partial t} + \frac{1}{2}\left|\frac{\partial u}{\partial x}\right|^2 + \frac{\partial^2 u}{\partial x^2} + \frac{\partial^4 u}{\partial x^4} = 0, \qquad x \in \Omega,\ t > 0, \tag{75}$$

where $u(x, t)$ is the scalar field. The dynamics are set by the competition among three terms: the second-order term $\partial^2 u/\partial x^2$ acts as a negative diffusion that injects energy and destabilizes long-wavelength modes; the fourth-order hyperviscous term $\partial^4 u/\partial x^4$ dissipates energy at short wavelengths and stabilizes the high-wavenumber modes; and the nonlinear term $\frac{1}{2}|\partial_x u|^2$ transfers energy across scales and couples the growing and decaying modes. This form is equivalent, up to a spatial derivative, to the more familiar advective form $u_t + u\,u_x + u_{xx} + u_{xxxx} = 0$: differentiating the equation above in x and writing $v = \partial_x u$ recovers $v_t + v\,v_x + v_{xx} + v_{xxxx} = 0$. The balance between the destabilizing second-order term and the stabilizing fourth-order term selects a band of linearly unstable wavenumbers, and once the domain is large enough to contain many such modes, the nonlinearity sustains an aperiodic, chaotic energy cascade among the resulting cellular structures.

Experiment Results: We present the complete experimental results in Table 33. As the data indicates, the agent exhibits significantly worse performance in this setting.

	RMSE	FRMSE
GNOT	5.674×10^{0}	5.862×10^{1}
Transolver	4.787×10^{0}	4.350×10^{1}
Transolver++	4.756×10^{0}	4.129×10^{1}
LNO	$\mathbf{2.807 \times 10^{0}}$	$\mathbf{3.569 \times 10^{1}}$
AMG	7.088×10^{0}	1.091×10^{2}
Agent	1.781×10^{1}	2.757×10^{2}

Table 33: Performance comparison on the **LongHorizon** dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 24 for visualizations of the dataset.

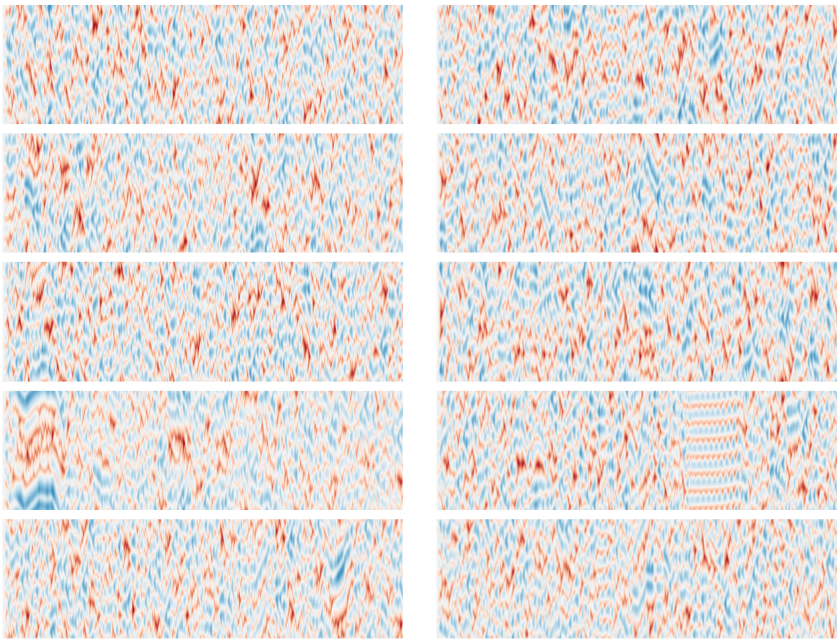


Figure 24: Visualization of the **MixedBC** dataset. The dataset contains solutions to the 1D Kuramoto-Sivashinsky equation. For visualization purposes, the x-axis represents time and the y-axis represents the spatial discretization grid.

I.19 STOCHASTIC

Dataset Description: This dataset contains two-dimensional stochastic Φ_2^4 dynamics evolving on a 128×128 spatial grid. The state is driven by the interplay of diffusion, a double-well nonlinearity, and space-time white noise. Unlike standard benchmarks where the future state is fully determined by the current one, the evolution here depends on the particular noise path realized over the prediction window, and this driving noise is provided to the model as additional input channels.

Modeling Objective: Let $x_t \in \mathbb{R}^{21\times128\times128}$ denote the model input at time t, formed by concatenating the current state $u_t \in \mathbb{R}^{1\times128\times128}$ with the driving noise increments $W_t, \ldots, W_{t+19} \in \mathbb{R}^{20\times128\times128}$ over the next 20 time steps along the channel dimension. Our objective is to learn a parameterized neural simulator Ψ such that $\hat{x}_{t+20} = \Psi(x_t)$, where the predicted output $\hat{x}_{t+20} \in \mathbb{R}^{1\times128\times128}$ is the state after the 20-step horizon. Conditioned on the complete noise path contained in the input, the solution map is deterministic, so the task is well-posed as pathwise regression.

Dataset Specific Challenges: The target state depends on the entire realized noise path, not just the current state. The model must resolve how each of the 20 stacked noise channels enters the dynamics at its respective time within the horizon and propagates through diffusion and the nonlinearity, effectively performing an implicit 20-step stochastic integration in a single forward pass.

Dataset Generation: The Φ_2^4 equation is discretized on a doubly-periodic 128×128 spatial grid over a square domain and advanced with an explicit finite-difference time-stepping scheme. The driving space-time white noise is synthesized on the grid from a truncated sine–cosine spectral basis: at each time slab an independent Gaussian field is drawn and band-limited at a spatial truncation scale ϵ, which regularizes the otherwise distributional noise so the explicit scheme is well-defined. Because the Φ_2^4 equation requires renormalization to admit a nontrivial continuum limit, we generate two versions of every trajectory: a renormalized solution, in which a mass counterterm depending on the truncation ϵ is subtracted to cancel the divergence induced by the noise, and a plain explicit solution without this correction. The initial condition is a fixed smooth sinusoidal field, $u_0(x, y) = \sin\big(2\pi(x+y)\big) + \cos\big(2\pi(x+y)\big)$, with an option to perturb it by a small random field for trajectory-to-trajectory variability.

Governing Physics: The dataset is governed by the two-dimensional stochastic quantization equation, the dynamic Φ_2^4 model, a semi-linear stochastic PDE in which linear diffusion competes with a double-well nonlinearity while being driven by space-time white noise. On the periodic domain $\Omega = [a, b]^2$ it is written as

$$\frac{\partial u}{\partial t} = \Delta u + \mu(u) + \sigma\,\xi(x,t), \qquad (x,t) \in \Omega \times (0,T], \tag{76}$$

where $u(x,t)$ is the scalar field, Δ is the two-dimensional Laplacian, ξ is space-time white noise (the distributional derivative of a cylindrical Wiener process), and σ scales the noise amplitude. The reaction term is the negative gradient of a symmetric double-well potential,

$$\mu(u) = 3u - u^3 = -\frac{\mathrm{d}V}{\mathrm{d}u}, \qquad V(u) = -\frac{3}{2}u^2 + \frac{1}{4}u^4, \tag{77}$$

so that the deterministic part of the drift pushes the field toward the two stable phases $u = \pm\sqrt{3}$ while diffusion penalizes sharp gradients, and the white noise perpetually perturbs the field between and around these wells. In two spatial dimensions the white-noise forcing is sufficiently rough that the cubic nonlinearity is ill-defined for the resulting distributional solutions; a renormalization procedure is therefore required, replacing the nonlinearity by a Wick-ordered counterpart in which a formally infinite, truncation-dependent mass counterterm is subtracted:

$$u^3 \;\longrightarrow\; u^3 - 3\,C_\epsilon\,u, \tag{78}$$

where the constant C_ϵ diverges as the noise truncation scale $\epsilon \to 0$. This counterterm is precisely what distinguishes the renormalized trajectories from the plain explicit ones, and it is what renders the Φ_2^4 dynamics well-posed as the discretization is refined.

Experiment Results: We present the experiment results in Table 34.

Visualization: We refer the readers to Figure 25 for visualizations of the dataset.

	RMSE	FRMSE
GNOT	3.990×10^{-3}	4.853×10^{-1}
Transolver	3.689×10^{-3}	4.495×10^{-1}
Transolver++	3.172×10^{-3}	3.818×10^{-1}
LNO	5.447×10^{-3}	6.771×10^{-1}
AMG	3.442×10^{-3}	4.141×10^{-1}
Agent	$\mathbf{2.324 \times 10^{-3}}$	$\mathbf{2.741 \times 10^{-1}}$

Table 34: Performance comparison on the `Stochastic` dataset. Best results are shown in **bold**.

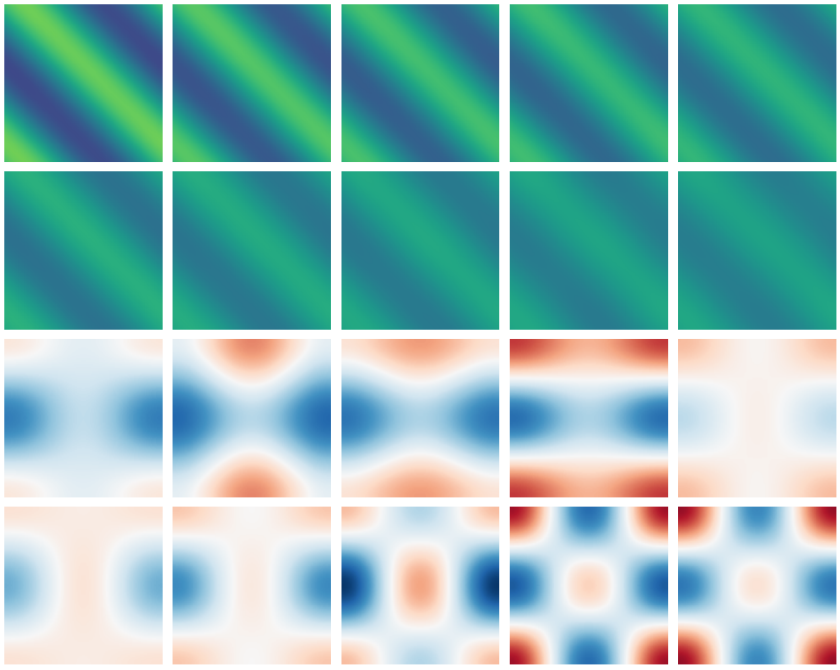


Figure 25: Visualization of the `Stochastic` dataset. This dataset contains solutions to the dynamical Φ^4_2 stochastic PDE. The first two rows display the renormalized solutions. The final two rows display the corresponding space-time white noise used to drive the system.

I.20 MULTIPHASE

Dataset Description: This dataset contains two-dimensional multi-phase Lagrangian particle simulations of liquid and gas confined in a closed box and driven by an external sloshing motion. Both phases are discretized as particles, labeled by a type indicator distinguishing liquid from gas, and per-type velocity statistics are used for normalization, reflecting the markedly different densities and motion scales of the two phases. The container undergoes a prescribed time-dependent excitation, described at each frame by linear and angular accelerations. Unlike single-phase Lagrangian benchmarks, the dynamics here arise from the interaction of two immiscible phases with a sharp, deforming interface, agitated by non-inertial forcing from the moving container.

Modeling Objective: Let $x_t \in \mathbb{R}^{N\times 55}$ denote the state of the N particles at time t, where each particle's 55-dimensional feature vector concatenates its current position in $\mathbb{R}^2$, a history of its 5 most recent velocities in $\mathbb{R}^{10}$ obtained by finite-differencing consecutive positions over the preceding frames, a particle type in $\{0,1\}$ indicating liquid or gas, and a sloshing feature in $\mathbb{R}^{42}$ stacking the 6-dimensional container accelerations over a 7-frame window covering the past 6 frames and the upcoming frame, broadcast identically to all particles. The number of particles N varies across samples, so batching is handled with a graph data loader that concatenates variable-sized particle sets. Our objective is to develop a neural simulator, defined as a parameterized function Ψ, that advances the system state by one time step. Formally, we aim to predict $\hat{v}_{t+1} = \Psi(x_t)$, where $\hat{v}_{t+1} \in \mathbb{R}^{N\times 2}$ is the per-particle, per-type normalized velocity over the next time step, from which positions are advanced during rollout.

Dataset Specific Challenges: The two phases carry strongly contrasting physical properties: the dense liquid dominates the momentum of the system while the light gas is easily displaced, and their interaction is mediated by a sharp interface that folds, breaks, and reconnects under sloshing. A single model must capture both phase behaviors and their coupling across the interface, using only the particle type label to differentiate them, while keeping the two phases from unphysically mixing over long rollouts.

Dataset Generation: This dataset is generated using a weakly-compressible Smoothed Particle Hydrodynamics (SPH) solver (Zhan et al., 2025; Domínguez et al., 2022). The simulation uses a particle spacing of $dp = 0.006$ m with a Wendland kernel and a smoothing length of $h = 0.91924\sqrt{2}\,dp$. Time integration employs a Symplectic scheme with a CFL number of 0.1, an initial timestep of 10^{-5} s, and a minimum timestep of 10^{-7} s, running to a physical time of 60 s with output written every 0.005 s. Multi-phase flow is enabled with the liquid phase modeled as water ($\rho_0 = 1000$ kg/m^3, $\gamma = 7$, $c_0 = 20$ m/s) and the gas phase as air ($\rho_0 = 1.18$ kg/m^3, $\gamma = 1.4$, $c_0 = 150$ m/s), each carrying a viscosity of 0.01. Viscosity is treated with the artificial formulation, and the Colagrossi–Landrini multiphase surface-tension coefficient is set to 0.1. A Molteni density diffusion term (coefficient 0.1) and particle shifting (coefficient -2, free-surface threshold 1.5) are applied to suppress numerical noise. The tank walls are fixed boundaries, and the sloshing motion is driven by a time-dependent acceleration field applied to the fluid, read from an external data file about an acceleration center at $(0.45, 0, 0)$ m with global gravity disabled for the forcing term. The domain is two-dimensional, spanning approximately $[-0.55, 0.55]$ m horizontally and $[-0.118, 0.62]$ m vertically, initialized with a lower water layer beneath an air region and containing 12,688 total particles.

Governing Physics: The sloshing case is governed by the two-phase weakly-compressible Navier–Stokes equations, in which two immiscible fluids—a heavy liquid (water) and a light gas (air)—coexist and interact across a sharp, deforming interface. Each phase $k \in \{\text{water}, \text{air}\}$ independently satisfies mass and momentum conservation, solved in Lagrangian form,

$$\frac{D\rho_k}{Dt} = -\rho_k \nabla\cdot\mathbf{u}, \qquad \frac{D\mathbf{u}}{Dt} = -\frac{1}{\rho_k}\nabla p_k + \nu_k \nabla^2\mathbf{u} + \mathbf{g}, \tag{79}$$

where ρ_k, p_k, and ν_k are the density, pressure, and kinematic viscosity of phase k, $\mathbf{u}$ is the velocity, and $\mathbf{g}$ is the body force (here the prescribed tank acceleration). The material derivative $D/Dt = \partial/\partial t + \mathbf{u}\cdot\nabla$ reflects the Lagrangian description, in which particles carry the properties of their own phase as they move; because each SPH particle is permanently tagged as either water or air, the interface is represented implicitly by the boundary between the two particle populations and requires no explicit interface tracking or reconstruction. Rather than enforcing strict incompressibility, each

phase is treated as weakly compressible and closed by its own Tait equation of state,

$$p_k = \frac{c_{0,k}^2\,\rho_{0,k}}{\gamma_k}\left[\left(\frac{\rho_k}{\rho_{0,k}}\right)^{\gamma_k} - 1\right], \tag{80}$$

where $\rho_{0,k}$ is the reference density, $c_{0,k}$ the artificial speed of sound, and γ_k the polytropic exponent *specific to that phase*. The defining feature of this dataset is the stark contrast between the two phases: the water phase uses $(\rho_0, \gamma, c_0) = (1000, 7, 20)$ while the air phase uses $(1.18, 1.4, 150)$, an $\mathcal{O}(10^3)$ density ratio and markedly different stiffness. This jump makes the pressure and density fields discontinuous across the interface, and the momentum coupling between the dense liquid and the compliant gas is what produces the buoyancy-driven sloshing, wave breaking, air entrainment, and splashing that characterize liquid–gas free-surface flows. To keep the two populations from artificially interpenetrating and to stabilize the pressure evaluation near this high-contrast interface, a multiphase surface correction is applied, and each phase's speed of sound $c_{0,k}$ is chosen large enough that its density fluctuations stay below roughly 1%, approximating incompressible behavior within each fluid while keeping the interfacial dynamics explicit.

Experiment Results: We present the complete experimental results in Table 35. While the main text reports the average RMSE of the liquid and gas phases, here we provide additional metrics disaggregated by phase to further analyze performance. We observe that the proposed agent architecture achieves superior results across all scenarios, demonstrating its overall effectiveness.

	RMSE (liquid)	LagKE (liquid)	Sinkhorn (liquid)	RMSE (gas)
GNOT	5.233×10^{-4}	6.509×10^{-4}	1.473×10^{-8}	1.123×10^{-3}
Transolver	5.787×10^{-4}	1.189×10^{-3}	4.566×10^{-8}	1.254×10^{-3}
Transolver+	5.471×10^{-4}	9.160×10^{-4}	2.803×10^{-8}	1.215×10^{-3}
LNO	5.496×10^{-4}	7.371×10^{-4}	8.130×10^{-9}	1.091×10^{-3}
AMG	5.500×10^{-4}	1.171×10^{-3}	3.026×10^{-8}	1.076×10^{-3}
Agent	$\mathbf{4.722 \times 10^{-4}}$	$\mathbf{5.109 \times 10^{-4}}$	$\mathbf{3.312 \times 10^{-9}}$	$\mathbf{1.044 \times 10^{-3}}$

	LagKE (gas)	Sinkhorn (gas)	EuARMS	EuAFRMSE
GNOT	9.569×10^{-3}	2.184×10^{-8}	3.617×10^{-4}	6.013×10^{-3}
Transolver	8.639×10^{-3}	5.797×10^{-8}	4.461×10^{-4}	7.180×10^{-3}
Transolver+	1.051×10^{-2}	4.384×10^{-8}	4.135×10^{-4}	6.866×10^{-3}
LNO	$\mathbf{1.001 \times 10^{-2}}$	2.446×10^{-8}	3.613×10^{-4}	6.041×10^{-3}
AMG	1.073×10^{-2}	2.058×10^{-8}	3.401×10^{-4}	5.838×10^{-3}
Agent	1.070×10^{-2}	$\mathbf{1.711 \times 10^{-8}}$	$\mathbf{3.066 \times 10^{-4}}$	$\mathbf{5.264 \times 10^{-3}}$

Table 35: Performance comparison on the **MultiPhase** dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 26 for visualizations of the dataset.



Figure 26: Visualization of the **MultiPhase** dataset. This dataset features Lagrangian particle data modeling multiphase liquid and gas dynamics within a closed container.

I.21 MIXEDPDE

Dataset Description: This dataset is a mixture of four two-dimensional PDE systems, each defined on a 128×128 spatial grid: advection-diffusion, Burgers' equation, Gray-Scott reaction-diffusion, and incompressible Navier-Stokes in vorticity form. The systems span qualitatively different dynamics, from linear transport and diffusive smoothing to nonlinear shock formation, pattern-forming chemical reactions, and turbulent vortex interactions. They also differ in their state dimensionality: advection-diffusion and Gray-Scott have a single physical channel, while Burgers' and Navier-Stokes have two.

Modeling Objective: Let $x_t \in \mathbb{R}^{C\times 128\times 128}$ denote the state of the system at time t, where the number of channels C depends on the underlying PDE ($C = 1$ for advection-diffusion and Gray-Scott, $C = 2$ for Burgers' and Navier-Stokes). The model additionally receives a PDE identifier $\mathrm{pde} \in \{0, 1, 2, 3\}$, indicating which of the four systems the trajectory belongs to (0: advection_diffusion_2d, 1: burgers_2d, 2: gray_scott_2d, 3: navier_stokes_vorticity_2d). Our objective is to develop a single neural simulator, defined as a parameterized function Ψ, that advances the system state by one time step. Formally, we aim to predict $\hat{x}_{t+1} = \Psi(x_t, \mathrm{pde})$, where the predicted output $\hat{x}_{t+1}$ retains the same shape $(C, 128, 128)$ as the input state.

Dataset Specific Challenges: A single model must represent four operators with fundamentally different governing physics, characteristic time scales, and channel semantics. Conditioning on the PDE identifier requires the solver to adapt its learned dynamics rather than average across systems, which typically degrades accuracy on all of them. Compounding this, the input and output channel dimensions vary across PDEs, so the model cannot rely on a fixed-size input/output projection; the architecture must handle variable channel counts natively, and batching becomes nontrivial since samples from different PDEs have incompatible tensor shapes.

Dataset Generation: These datasets are generated using Exponential Time Differencing (ETD) Runge–Kutta spectral schemes. All fields are discretized on a 128×128 doubly-periodic grid and advanced with a fixed timestep, where the linear terms are integrated exactly in Fourier space and the nonlinear terms are treated with an ETDRK time-stepping scheme. For each PDE we generate 30 trajectories of 100 steps each, producing arrays of shape $(30, 101, C, 128, 128)$ where C is the number of physical channels. Initial conditions are drawn from a random truncated Fourier series (independent per channel, with a low wavenumber cutoff and amplitude normalized to $[-1, 1]$), except for the Gray–Scott system, which is initialized with $u \approx 1$ and $v \approx 0$ perturbed by up to eight randomly placed Gaussian blobs (dipping u toward 0.5 and raising v toward 0.25) plus small symmetry-breaking noise. Per-PDE solver settings are: Advection–Diffusion (domain extent 1.0, $\Delta t = 0.01$, velocity $(0.4, 1.0)$, diffusivity 10^{-3}, single channel); Burgers (domain extent 1.0, $\Delta t = 0.01$, diffusivity 10^{-2}, two velocity channels); Navier–Stokes in vorticity form (domain extent 1.0, $\Delta t = 0.01$, diffusivity 3×10^{-4}, single scalar-vorticity channel, with 10 inner substeps per saved frame); and Gray–Scott (domain extent 1.0, $\Delta t = 1.0$, diffusivities 2×10^{-5} and 10^{-5}, feed rate 0.04, kill rate 0.06, two species channels, with 50 inner substeps per saved frame).

Governing Physics: The four datasets are governed by distinct semi-linear PDEs on the two-dimensional periodic domain, each written in the form $\partial_t \mathbf{u} = \mathcal{L}\mathbf{u} + \mathcal{N}(\mathbf{u})$ with a linear operator $\mathcal{L}$ integrated exactly in spectral space and a nonlinear term $\mathcal{N}$. The advection–diffusion equation transports a passive scalar u at constant velocity $\mathbf{c}$ while it diffuses,

$$\frac{\partial u}{\partial t} + \mathbf{c} \cdot \nabla u = \nu \, \Delta u, \tag{81}$$

where ν is the diffusivity; this case is purely linear. The Burgers equation adds nonlinear self-advection to the same diffusive structure, here for a vector velocity field $\mathbf{u}$,

$$\frac{\partial \mathbf{u}}{\partial t} + \tfrac{1}{2}\nabla \cdot (\mathbf{u} \otimes \mathbf{u}) = \nu \, \Delta \mathbf{u}, \tag{82}$$

whose quadratic convection steepens gradients into sharp fronts balanced by viscous dissipation. The incompressible Navier–Stokes equations are solved in the vorticity formulation, where the scalar vorticity $\omega = \nabla \times \mathbf{u}$ is advected by the velocity field it induces,

$$\frac{\partial \omega}{\partial t} + (\mathbf{u} \cdot \nabla)\, \omega = \nu \, \Delta \omega, \qquad \mathbf{u} = \nabla^{\perp} \Delta^{-1} \omega, \tag{83}$$

with the velocity recovered from vorticity through a streamfunction, producing two-dimensional turbulence. Finally, the Gray–Scott model is a two-species reaction–diffusion system in which chemicals u and v diffuse at different rates and react nonlinearly,

$$\frac{\partial u}{\partial t} = D_u \, \Delta u - uv^2 + F(1-u), \qquad \frac{\partial v}{\partial t} = D_v \, \Delta v + uv^2 - (F+\kappa)\, v, \tag{84}$$

where D_u, D_v are the species diffusivities, F the feed rate, and κ the kill rate; the cubic autocatalytic coupling uv^2 together with differential diffusion drives the spontaneous formation of spots, stripes, and self-replicating patterns. Across all four systems the exact spectral treatment of the linear operator $\mathcal{L}$ and periodic boundaries are what make the ETDRK integrator stable and accurate at the chosen timesteps.

Experiment Results: We present the complete experimental results in Table 36, where we additionally report the individual RMSE for each of the four PDEs. The results demonstrate that the agent proposed architecture achieves strong performance across all evaluated aspects.

	RMSE	FRMSE	RMSE (Adv Diff)	RMSE (Burgers)
GNOT	8.792×10^{-2}	6.037×10^{0}	1.944×10^{-1}	2.214×10^{-2}
Transolver	7.565×10^{-2}	4.685×10^{0}	1.745×10^{-1}	2.194×10^{-2}
Transolver++	7.535×10^{-2}	4.352×10^{0}	1.736×10^{-1}	2.131×10^{-2}
LNO	7.684×10^{-2}	4.037×10^{0}	1.761×10^{-1}	2.199×10^{-2}
AMG	8.845×10^{-2}	6.221×10^{0}	1.922×10^{-1}	2.230×10^{-2}
Agent	$\mathbf{2.701 \times 10^{-2}}$	$\mathbf{2.264 \times 10^{0}}$	$\mathbf{2.256 \times 10^{-3}}$	$\mathbf{6.041 \times 10^{-3}}$

	RMSE (Gray Scott)	RMSE (NS)
GNOT	4.709×10^{-2}	5.193×10^{-2}
Transolver	4.515×10^{-2}	2.119×10^{-2}
Transolver++	4.562×10^{-2}	2.174×10^{-2}
LNO	5.238×10^{-2}	1.907×10^{-2}
AMG	6.176×10^{-2}	5.106×10^{-2}
Agent	$\mathbf{4.630 \times 10^{-2}}$	$\mathbf{3.797 \times 10^{-3}}$

Table 36: Performance comparison on the **MixedPDE** dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 27 for visualizations of the dataset.

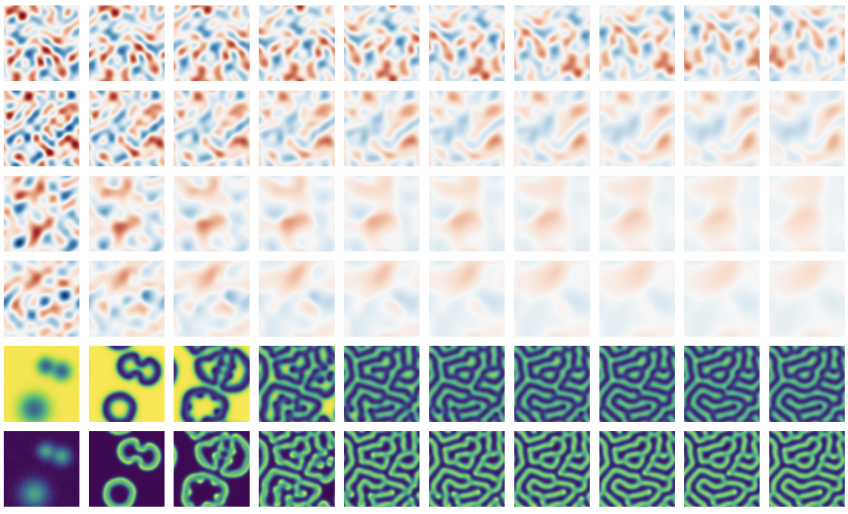


Figure 27: Visualization of the **MixedPDE** dataset. This dataset contains a mixture of 2D PDEs. The first row corresponds to advection-diffusion, while the second row displays Navier-Stokes vorticity. The third and fourth rows represent the two channels of the Burgers' equation, and the final two rows depict the two channels of the Gray-Scott model.

I.22 LARGELAG

Dataset Description: This dataset contains large-scale Lagrangian simulations of flow around a cylinder. Both the fluid and the solid cylinder are discretized into small particles, with more than 2 million particles per frame. The simulation is two-dimensional, and the fluid domain is open: particles enter at the inlet and leave at the outlet.

Modeling Objective: Let $x_t \in \mathbb{R}^{N\times 13}$ denote the state of the system at time t, where N is the number of particles in the current window. The 13 input features per particle are the last-step position (2), the historical velocities over the previous 5 steps ($2\times 5 = 10$), and a scalar particle-type flag distinguishing fluid from solid (1). Our objective is to develop a neural simulator, defined as a parameterized function Ψ, that advances the system state by one time step. Formally, we aim to predict $\hat{x}_{t+1} = \Psi(x_t)$, where $\hat{x}_{t+1} \in \mathbb{R}^{N\times 2}$ is the per-particle velocity at the next time step. Solid particles have zero velocity by construction and are excluded from the loss, as are fluid particles that have left the computational domain by the target frame.

Dataset Specific Challenges: The sheer scale of the particle set makes memory the dominant constraint. Any model must be memory efficient, since methods whose cost grows quadratically with the number of particles, such as dense global attention or dense neighborhood graphs, are infeasible at this resolution.

Dataset Generation: This dataset is generated using a weakly-compressible Smoothed Particle Hydrodynamics (SPH) solver (Zhan et al., 2025; Domínguez et al., 2022), in a two-dimensional open-domain configuration with inflow/outflow boundaries. The simulation uses a particle spacing of $dp = 0.0015$ m with a Wendland kernel and a smoothing length of $h = 2\,dp$. Time integration employs a Symplectic scheme with a CFL number of 0.2, running to a physical time of 10 s with output written every 0.005 s. The fluid has reference density $\rho_0 = 1000$ kg/m^3, $\gamma = 7$, and an artificial speed of sound of 15 m/s. Viscosity is treated with the Laminar+SPS formulation at a kinematic viscosity of 4×10^{-5} m^2/s. A Fourtakas density diffusion term and full particle shifting are applied to suppress numerical noise, and boundaries use the modified Dynamic Boundary Condition (mDBC) with corrected wall normals. A uniform inflow of 1.0 m/s is imposed at the left open boundary (with entering fluid removed and re-created over four buffer layers), while the right open boundary acts as an outlet that converts fluid and extrapolates density from ghost nodes. The channel is bounded above and below by solid walls translating with the flow, and a rigid cylinder of radius 0.1 m is placed in the stream as the bluff body. The domain spans roughly $[-0.6, 2.1]$ m streamwise and $[-1.03, 1.03]$ m cross-stream, discretized into approximately 2.31 million total particles.

Governing Physics: The flow past the cylinder is governed by the weakly-compressible Navier–Stokes equations, solved in Lagrangian form where the fluid satisfies mass and momentum conservation,

$$\frac{D\rho}{Dt} = -\rho\,\nabla\cdot\mathbf{u}, \qquad \frac{D\mathbf{u}}{Dt} = -\frac{1}{\rho}\nabla p + \nu\,\nabla^2\mathbf{u}, \tag{85}$$

where ρ is density, $\mathbf{u}$ the velocity, p the pressure, and ν the kinematic viscosity; gravity is absent, so the dynamics are driven purely by the imposed free stream and its interaction with the obstacle. The material derivative $D/Dt = \partial/\partial t + \mathbf{u}\cdot\nabla$ reflects the Lagrangian description, in which particles carry the fluid properties as they move. Rather than enforcing strict incompressibility, the fluid is treated as weakly compressible and closed by the Tait equation of state,

$$p = \frac{c_0^2\,\rho_0}{\gamma}\left[\left(\frac{\rho}{\rho_0}\right)^{\gamma} - 1\right], \tag{86}$$

where ρ_0 is the reference density, c_0 the artificial speed of sound, and γ the polytropic exponent; c_0 is chosen large enough that density fluctuations stay below roughly 1%, approximating incompressible behavior while keeping the pressure evaluation explicit. The defining feature of this case is the rigid circular cylinder of diameter D immersed in the uniform stream U_∞: the cylinder enforces a no-slip, no-penetration condition on its surface, forcing the incoming flow to separate from the curved wall.

Experiment Results: We present the complete experimental results in Table 37. We omit the Sinkhorn divergence and Eulerian aggregated metrics, as their computation becomes prohibitively expensive for a system of two million particles.

Visualization: We refer the readers to Figure 28 for visualizations of the dataset.

	RMSE	Kinetic Energy
GNOT	1.300×10^{-2}	9.454×10^{3}
Transolver	2.753×10^{-2}	8.507×10^{4}
Transolver++	2.027×10^{-2}	4.624×10^{4}
LNO	1.599×10^{-1}	7.085×10^{5}
AMG	–	–
Agent	$\mathbf{5.666 \times 10^{-3}}$	$\mathbf{7.876 \times 10^{3}}$

Table 37: Performance comparison on the **LargeLag** dataset. Best results are shown in **bold**.

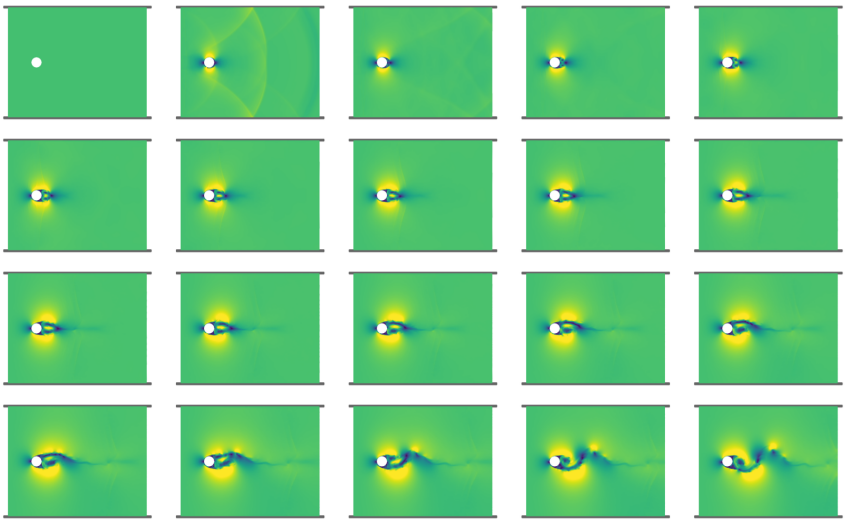


Figure 28: Visualization of the **LargeLag** dataset. This dataset contains a high-fidelity, large-scale Lagrangian simulation of flow around a cylinder with over 2 million particles.

I.23 SIM2REAL

Dataset Description: This dataset contains fluid dynamics of vortex-induced vibration in a tandem-cylinder configuration. Two cylinders are arranged in tandem, one positioned downstream of the other, so that the wake shed by the upstream cylinder impinges on the downstream body. This coupling produces complex vortex-shedding dynamics and flow-induced structural vibration that neither cylinder would exhibit in isolation.

Modeling Objective: Let $x_t \in \mathbb{R}^{2\times 128\times 128}$ denote the state of the system at time t. The two feature channels represent the horizontal velocity and vertical velocity fields, each evaluated on a 128×128 spatial discretization grid. Our objective is to develop a neural simulator, defined as a parameterized function Ψ, that advances the system state by one time step. Formally, we aim to predict $\hat{x}_{t+1} = \Psi(x_t)$, where the predicted output $\hat{x}_{t+1}$ retains the same spatial and feature dimensions $(2, 128, 128)$ as the input state.

Dataset Specific Challenges: This dataset follows a simulation-to-real (sim2real) setup: the model is trained on simulated data but evaluated on real-world measurements. The distribution shift between simulated and real flow fields means that a model fitting the training data well may still generalize poorly at test time. Moreover, real-world data carries measurement noise and unmodeled physical effects absent from the simulation, so a neural simulator must capture the underlying flow dynamics rather than simulation-specific artifacts in order to remain accurate and temporally stable when rolled out on real data.

Dataset Generation: We adopt the fluid–structure interaction (FSI) dataset from Hu et al. (2026), using its paired numerical trajectories. The simulated data are produced with a two-dimensional incompressible finite-volume flow solver on a uniform Cartesian grid that represents the immersed cylinder through the Boundary Data Immersion Method (BDIM), a second-order immersed-boundary treatment that avoids body-fitted remeshing as the structure moves. The rigid cylinder is not held fixed but is elastically mounted: it is free to vibrate under the instantaneous fluid loading, and its structural response is advanced by a spring–mass–damper equation integrated in lockstep with the flow solver, so that the fluid and body states are updated in a two-way-coupled manner at every timestep.

Governing Physics: The FSI case is governed by a two-way coupling between an incompressible fluid and an elastically mounted rigid cylinder. The fluid obeys the incompressible Navier–Stokes equations,

$$\nabla \cdot \mathbf{u} = 0, \qquad \frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} = -\frac{1}{\rho}\nabla p + \nu\,\nabla^2\mathbf{u}, \tag{87}$$

with velocity $\mathbf{u}$, pressure p, density ρ, and kinematic viscosity ν. The distinguishing element is the cylinder itself: unlike the stationary-cylinder wake, here the circular cylinder is a movable structure suspended on a spring and damper, and its center displacement $\mathbf{x}_s(t)$ evolves according to a forced oscillator (spring–mass–damper) equation,

$$m\,\ddot{\mathbf{x}}_s + c\,\dot{\mathbf{x}}_s + k\,\mathbf{x}_s = \mathbf{F}(t), \tag{88}$$

where m, c, and k are the oscillating mass, damping, and stiffness of the mount, and $\mathbf{F}(t)$ is the hydrodynamic force the flow exerts on the cylinder, obtained by integrating the pressure and viscous stress over its wetted surface Γ_{cyl},

$$\mathbf{F}(t) = \oint_{\Gamma_{\text{cyl}}} \left[-p\,\mathbf{I} + \mu\left(\nabla\mathbf{u} + \nabla\mathbf{u}^{\top}\right)\right]\cdot\mathbf{n}\,\mathrm{d}S. \tag{89}$$

The two systems are locked together by the no-slip, no-penetration condition on the moving cylinder wall,

$$\mathbf{u} = \dot{\mathbf{x}}_s \quad \text{on } \Gamma_{\text{cyl}}, \tag{90}$$

so the cylinder both stirs the fluid and is pushed by it. The character of this coupled response is fixed by three dimensionless groups built around the cylinder: the Reynolds number $\mathrm{Re} = UD/\nu$ (with D the cylinder diameter), the mass ratio $m^* = m/(\rho\,\frac{\pi}{4}D^2)$ comparing structural to displaced-fluid inertia, and the damping ratio $\zeta = c/(2\sqrt{km})$.

Experiment Results: We present the complete experimental results in Table 38. The findings indicate that the agent proposed architecture exhibits poor performance in this setting.

Visualization: We refer the readers to Figure 29 for visualizations of the dataset.

	RMSE	FRMSE	Kinetic Energy
GNOT	1.987×10^{-3}	1.739×10^{-1}	1.474×10^{-4}
Transolver	$\mathbf{1.783 \times 10^{-3}}$	$\mathbf{1.535 \times 10^{-1}}$	1.432×10^{-4}
Transolver++	1.809×10^{-3}	1.568×10^{-1}	$\mathbf{1.430 \times 10^{-4}}$
LNO	8.306×10^{-3}	9.829×10^{-1}	1.035×10^{-3}
AMG	2.705×10^{-3}	2.424×10^{-1}	2.257×10^{-4}
Agent	6.734×10^{-3}	6.740×10^{-1}	5.157×10^{-4}

Table 38: Performance comparison on the **Sim2Real** dataset. Best results are shown in **bold**.

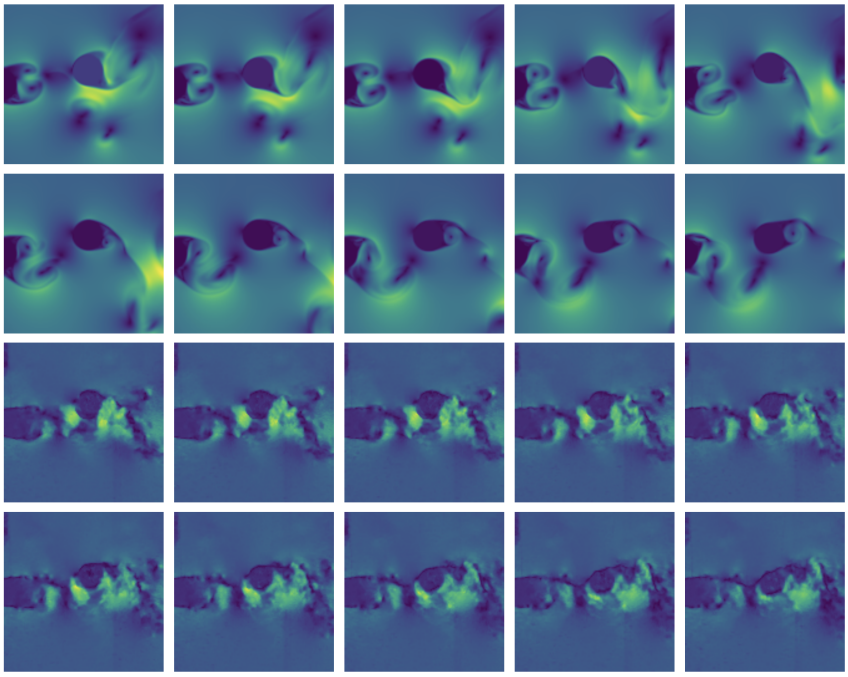


Figure 29: Visualization of the **Sim2Real** dataset. The top two rows present numerically simulated data for vortex-induced vibration in a tandem-cylinder arrangement, while the bottom two rows display real-world experimental data.

I.24 HighDimensional

Dataset Description: This dataset contains gyrokinetic simulations of plasma micro-turbulence in a magnetized fusion plasma. The system evolves the perturbed gyrocenter distribution function on a five-dimensional phase-space grid comprising two velocity-space coordinates, the parallel velocity $v_\parallel$ and the magnetic moment μ, and three spatial coordinates: the position s along the magnetic field line and the two perpendicular directions represented spectrally by wavenumbers k_x and k_y. As the turbulence develops, fine-scale structures in the distribution function couple across velocity space, the field-aligned direction, and the perpendicular spectral modes. Each trajectory records 266 time steps of the turbulent dynamics.

Modeling Objective: Let $x_t \in \mathbb{R}^{2\times32\times8\times16\times85\times32}$ denote the state of the system at time t. The dimensions are $(2, n_{v_\parallel}, n_\mu, n_s, n_{k_x}, n_{k_y})$, where the leading channel dimension of size 2 holds the real and imaginary parts of the spectral distribution function. Our objective is to develop a neural simulator, defined as a parameterized function Ψ, that advances the system state by one time step. Formally, we aim to predict $\hat{x}_{t+1} = \Psi(x_t)$, where the predicted output $\hat{x}_{t+1}$ retains the same shape $(2, 32, 8, 16, 85, 32)$ as the input state.

Dataset Specific Challenges: The state lives on a five-dimensional phase-space grid rather than the 2D or 3D spatial grids for which most neural PDE surrogates are designed, with a single state containing roughly 22.3 million degrees of freedom. The five axes are physically heterogeneous: two are velocity-space coordinates, one is a field-aligned spatial coordinate, and two are spectral (Fourier) coordinates that carry no translational structure in the usual convolutional sense. Consequently, standard architectural assumptions such as 2D convolutions, isotropic attention over pixels, and spatial FFT layers do not transfer directly and must be rethought for mixed position/velocity/spectral axes. Compounding this, the data is too large for in-memory processing, since each state occupies roughly 85 MB in float32, so training demands small batch sizes and memory-efficient architectures.

Dataset Generation: The trajectories are produced with GKW, a nonlinear gyrokinetic flux-tube code that solves the perturbed 5D gyrokinetic equation. To keep the cost of generating a large dataset manageable, the simulations use the adiabatic-electron approximation, so only ion-temperature-gradient-driven turbulence is evolved and the electron distribution need not be solved; each nonlinear run is carried out on a 5D grid of resolution over the two velocity-space coordinates (parallel velocity $v_\parallel$ and magnetic moment μ) and three spatial coordinates (the field-line coordinate s, the radial direction x, and the binormal direction y), with the field stored spectrally in (k_x, k_y) and each snapshot carrying two channels for the real and imaginary parts of the ballooning transform. Four operating parameters are varied to span turbulent regimes—the ion temperature gradient $R/L_T \in [1, 12]$, the density gradient $R/L_n \in [1, 7]$, the safety factor $q \in [1, 9]$, and the magnetic shear $\hat{s} \in [0.5, 5]$. Each simulation is integrated for $31{,}920$ steps, averaged every 40 steps and subsampled every third step to yield 266 snapshots.

Governing Physics: The dataset is governed by the nonlinear gyrokinetic equation, the reduced kinetic description of magnetized-plasma turbulence. Starting from the 6D Vlasov equation for the distribution function $f(\mathbf{r}, \mathbf{v}, t)$ and transforming to guiding-center coordinates $(\mathbf{R}, v_\parallel, \mu, \theta)$, one exploits the scale separation between the Larmor radius ρ and the background gradient length L ($\rho \ll L$) and the low-frequency ordering (frequencies well below the cyclotron frequency) to gyroaverage over the gyrophase θ. This removes one velocity dimension and leaves a 5D distribution function $f = f(k_x, k_y, s, v_\parallel, \mu)$ evolving over time. Its perturbed evolution, for each species, takes the split form

$$\frac{\partial f}{\partial t} + (v_\parallel \mathbf{b} + \mathbf{v}_D) \cdot \nabla f - \frac{\mu B}{m} \frac{\mathbf{B} \cdot \nabla B}{B^2} \frac{\partial f}{\partial v_\parallel} + \mathbf{v}_\chi \cdot \nabla f = S, \tag{91}$$

where $\mathbf{b} = \mathbf{B}/B$ is the unit vector along the magnetic field, $\mathbf{v}_D$ is the magnetic drift arising from gradients and curvature in $\mathbf{B}$, S collects sources and collisions, and $\mathbf{v}_\chi = \frac{c}{B}\,\mathbf{b} \times \nabla\chi$ is the drift induced by the gyroaveraged generalized potential $\chi = \langle \phi - \frac{v_\parallel}{c} A_\parallel \rangle$. The nonlinear term $\mathbf{v}_\chi \cdot \nabla f$ is the $\mathbf{E} \times \mathbf{B}$ advection of the distribution by its own self-consistent potential: it is the central driver of turbulence and the most expensive term to evaluate, and it is precisely the physics that reduced (quasilinear) models omit. The system is closed self-consistently, with the electrostatic potential

recovered from a velocity-space integral of f through the gyroaveraging Bessel factor J_0,

$$\phi = A \int J_0 \, f \, \mathrm{d}v_{\parallel} \, \mathrm{d}\mu, \tag{92}$$

so that the field feeding $\mathbf{v}_\chi$ is itself a moment of the state it advects. In practice the distribution is split into an equilibrium and a perturbation, $f = f_0 + \delta f$, with δf separating into a passive adiabatic response $-\frac{Z\phi}{T} f_0$ and a non-adiabatic kinetic part that carries the irreversible, transport-producing dynamics; the adiabatic-electron approximation used here treats the electron response as purely adiabatic, leaving only the ion kinetic equation to be solved. The physically decisive quantity is the radial heat flux transported toward the reactor wall,

$$Q = \int_C \int_v 2 \, \phi \, f \, \mathrm{d}v_{\parallel} \, \mathrm{d}\mu \, \mathrm{d}x \, \mathrm{d}y \, \mathrm{d}s. \tag{93}$$

A simulation first passes through a linear phase in which unstable k_y modes grow, then saturates into a statistically steady turbulent state regulated by zonal flows—the $k_y = 0$ component of ϕ—which shear apart eddies and set the final transport level.

Experiment Results: The complete experimental results are detailed in Table 39. For this analysis, RMSE serves as the only evaluation metric. Furthermore, due to the high dimensionality of the 5D data, there are no general-purpose baselines available for comparison.

	RMSE
GNOT	–
Transolver	–
Transolver++	–
LNO	–
AMG	–
Agent	$\mathbf{3.412 \times 10^{-4}}$

Table 39: Performance comparison on the `HighDimensional` dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 30 for visualizations of the dataset.

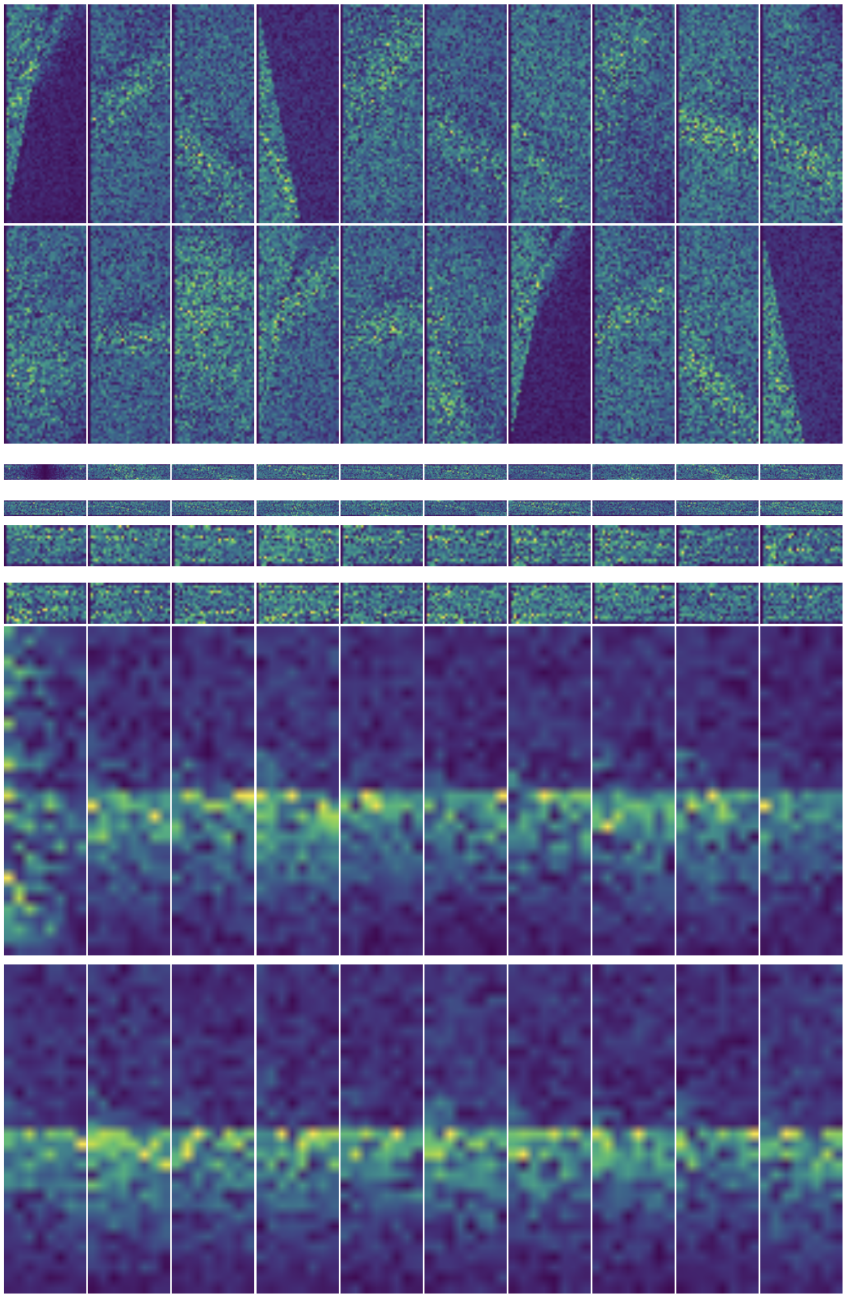


Figure 30: Visualization of the **HighDimensional** dataset. This dataset contains high-dimensional gyrokinetic PDE data, represented as a complex-valued 5D array of the form `(2, nvpar, nmu, ns, nkx, nky)`. The first dimension denotes the real and imaginary components. The variables `nvpar` and `nmu` represent the two velocity-space coordinates (parallel velocity, $v_\parallel$, and magnetic moment, μ), while `ns`, `nkx`, and `nky` represent the three spatial coordinates: the position s along the magnetic field line and the two perpendicular directions represented spectrally by wavenumbers. In the figure, the first two rows display the `kx_ky` cross-section, the next two show the `s_kx` cross-section, followed by two rows for `s_ky`, and the final two rows displaying the `vpar_mu` cross-section.

I.25 ComplexGeometry

Dataset Description: This dataset contains steady-state aerodynamic simulations of turbulent airflow around 3D car geometries, derived from the ShapeNet "car" category. Each sample is an automobile shape immersed in an incompressible fluid stream. As air flows over and around the vehicle, it produces a pressure distribution across the car surface and a surrounding velocity field in the volume, including a downstream wake. The data is provided as an unstructured point cloud: exterior (volume) points sampled around the car and surface points sampled on the car body, with connectivity given as a graph.

Modeling Objective: Let $x \in \mathbb{R}^{N\times 7}$ denote the input features for a single car, where N is the number of points (volume and surface). The seven input channels are the 3D point position (x, y, z), the signed distance to the car surface (sdf), and the 3D surface normal (n_x, n_y, n_z). Our objective is to develop a neural simulator, defined as a parameterized function Ψ, that predicts the flow fields $\hat{y} = \Psi(x)$, where $\hat{y} \in \mathbb{R}^{N\times 4}$ has four output channels per point: the 3D fluid velocity (v_x, v_y, v_z) and the pressure (p).

Dataset Specific Challenges: The domain is a large, irregular 3D point cloud rather than a regular grid, with points spanning both the thin car surface and the surrounding volume at very different densities. Models must therefore handle this unstructured, multi-scale geometry while respecting the sharp distinction between on-surface and off-surface behavior.

Dataset Generation: We adopt the ShapeNet-Car dataset of Umetani & Bickel (2018), whose car geometries are drawn from the "car" category of ShapeNet (Chang et al., 2015). The flow around each car is computed with OpenFOAM, using its steady-state incompressible solver based on the SIMPLE algorithm (a segregated pressure–velocity coupling that iterates the Reynolds-averaged equations to a converged steady state) with a finite-volume discretization on an unstructured mesh. Each car is placed in a virtual wind tunnel with a uniform inflow of 20 m/s (i.e. the 72 km/h driving condition), a no-slip condition imposed on the car surface, and Reynolds-averaged turbulence closure to represent the effect of unresolved turbulent fluctuations. Overall, 889 samples with distinct car shapes are simulated; the domain is discretized into an unstructured mesh with 32,186 mesh points, and each simulation records the air flow field (velocity and pressure) in the volume surrounding the car together with the pressure distribution over the car surface.

Governing Physics: The ShapeNet-Car case is governed by the steady, incompressible Reynolds-Averaged Navier–Stokes (RANS) equations for the airflow around the vehicle. Decomposing the velocity into a mean and a fluctuating part, $\mathbf{u} = \bar{\mathbf{u}} + \mathbf{u}'$, and time-averaging the incompressible Navier–Stokes equations gives

$$\nabla \cdot \bar{\mathbf{u}} = 0, \qquad (\bar{\mathbf{u}} \cdot \nabla)\, \bar{\mathbf{u}} = -\frac{1}{\rho}\nabla \bar{p} + \nu\, \nabla^2 \bar{\mathbf{u}} + \nabla \cdot \boldsymbol{\tau}^R, \tag{94}$$

where $\bar{\mathbf{u}}$ and $\bar{p}$ are the mean velocity and pressure, ρ the air density, and ν the kinematic viscosity. The time-averaging leaves an extra term, the Reynolds stress $\boldsymbol{\tau}^R = -\overline{\mathbf{u}' \otimes \mathbf{u}'}$, which represents the mean momentum transport by turbulent fluctuations and must be modeled to close the system. Under the Boussinesq eddy-viscosity hypothesis it is related to the mean strain rate through a turbulent viscosity ν_t,

$$-\overline{u_i' u_j'} = \nu_t \left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i} \right) - \frac{2}{3} k\, \delta_{ij}, \tag{95}$$

where k is the turbulent kinetic energy and ν_t is supplied by a RANS turbulence model. The defining feature of this benchmark is the car itself as a bluff body immersed in the uniform stream U_∞: the no-slip condition on the vehicle surface forces the flow to develop thin boundary layers, separate over the rear and other curved regions, and leave a turbulent wake, and it is this flow–geometry interaction that sets the pressure field on the body. The quantity of engineering interest is the aerodynamic drag, obtained by integrating the surface pressure and viscous (skin-friction) stress over the car surface S and projecting onto the streamwise direction $\mathbf{e}_x$,

$$F_d = \oint_S \left[-\bar{p}\, \mathbf{n} + \boldsymbol{\tau}_w\right] \cdot \mathbf{e}_x \, \mathrm{d}S, \qquad C_d = \frac{F_d}{\frac{1}{2}\rho\, U_\infty^2\, A}, \tag{96}$$

where $\boldsymbol{\tau}_w$ is the wall-shear stress, A a reference frontal area, and C_d the drag coefficient. Because C_d depends on the full separated, turbulent flow established by the car's shape, accurately predict-

ing it requires capturing how geometry variations reshape the surface pressure and wake—the core physical challenge the dataset poses.

Experiment Results: We present the complete experimental results in Table 40. In addition to the volumetric RMSE reported in the main text, we provide the surface RMSE and the RMSE of the drag coefficient. The results indicate that the agent proposed architecture exhibits sub-optimal performance in this setting.

	RMSE	RMSE (surface)	RMSE (drag)
GNOT	$\mathbf{7.638 \times 10^{-2}}$	$\mathbf{1.741 \times 10^{-1}}$	6.132×10^{-3}
Transolver	8.295×10^{-2}	1.785×10^{-1}	$\mathbf{5.690 \times 10^{-3}}$
Transolver++	9.702×10^{-2}	2.044×10^{-1}	9.061×10^{-3}
LNO	1.032×10^{-1}	2.058×10^{-1}	9.151×10^{-3}
AMG	1.448×10^{-1}	2.728×10^{-1}	1.855×10^{-2}
Agent	1.366×10^{-1}	2.729×10^{-1}	1.855×10^{-2}

Table 40: Performance comparison on the `ComplexGeometry` dataset. Best results are shown in **bold**.

Visualization: We refer the readers to Figure 31 for visualizations of the dataset.

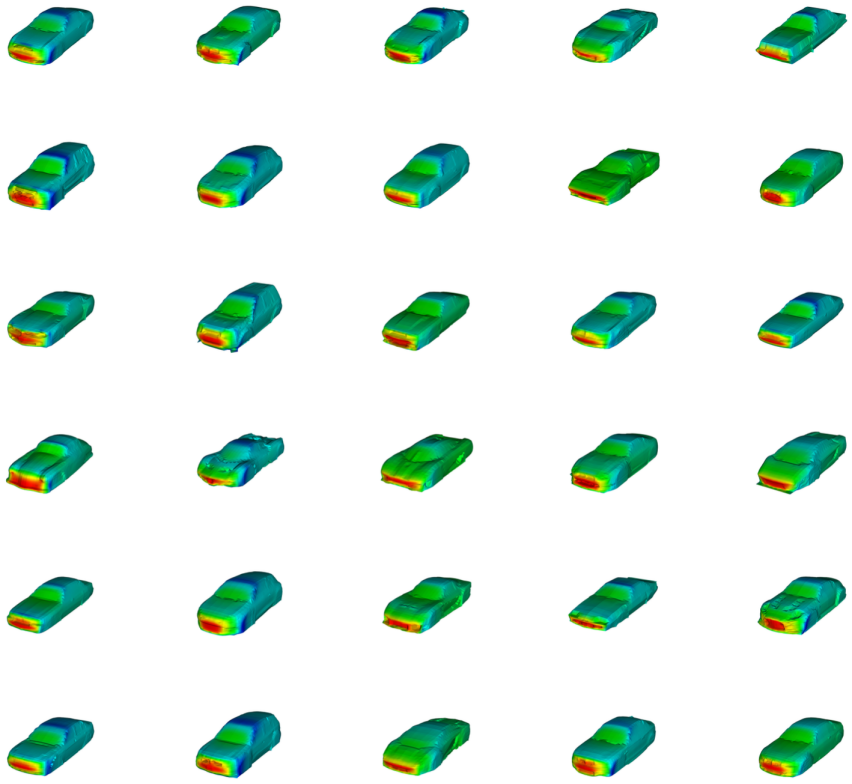


Figure 31: Visualization of the `ComplexGeometry` dataset. This dataset comprises steady-state aerodynamic simulations of 3D car models. he visualizations show surface pressure, and the full data also includes volumetric velocity.