Diagnosing Failure Modes of Neural Operators Across Diverse PDE Families

Anonymous authors
Paper under double-blind review

Abstract

Neural PDE solvers are increasingly used as learned surrogates for families of partial differential equations, where the key machine learning challenge is not only interpolation on a fixed benchmark distribution but generalization under structured shifts in coefficients, boundary conditions, discretization, and rollout horizon. Yet evaluation is still often dominated by in-distribution test error, making robustness difficult to assess. We introduce a standardized stress-testing framework for neural PDE solvers under deployment-relevant shift. We instantiate it on three representative architectures—Fourier Neural Operators (FNOs), a DeepONet-style model, and convolutional neural operators (CNOs)—across five qualitatively different PDE families: dispersive, elliptic, multi-scale fluid, financial, and chaotic systems. Across 750 trained models, we measure robustness using baseline-normalized degradation factors together with spectral and rollout diagnostics. The resulting comparisons reveal that strong in-distribution accuracy does not reliably predict robustness, and that failure patterns depend jointly on architecture and PDE family. Our results provide a clearer basis for evaluating robustness claims in neural PDE solvers and suggest that function-space generalization under structured shift should be treated as a first-class evaluation target.

1 Introduction

Neural PDE solvers learn operators between function spaces, offering fast surrogates for families of partial differential equations rather than repeated solves of individual discretized instances [7; 8]. Architectures such as FNO [14], DeepONet [17], and CNO [23], along with wavelet-, geometry-, and attention-based variants, have achieved strong benchmark performance [29; 15; 13]. But for machine learning, low matched-distribution test error is not the whole problem. The real question is whether these models generalize under the structured shifts that arise in use: changes in coefficients, boundary or terminal conditions, discretization, and rollout horizon.

That question remains under-studied. Current evaluation is still dominated by in-distribution error, limited perturbation analyses, or single-equation benchmarks [27; 5; 20]. This is a serious limitation because recent theory suggests that different neural-operator architectures should fail in different ways. For FNO-style models, robustness depends on truncation, discretization, spectral structure, and trainability [3; 11; 26; 6]. For DeepONet-style models, optimization, generalization, and model size matter, and linear-reconstruction methods can be fundamentally inefficient on discontinuous operators [12; 19; 10]. Boundary variation can also change the effective operator family rather than merely induce a mild test perturbation [25; 18]. Similar concerns appear elsewhere in scientific ML, including documented PINN failure modes and spectral bias under high-frequency or multiscale structure [9; 28; 2].

We address this gap through a stress-testing framework for neural PDE solvers under structured shift. We evaluate three representative architectures—FNO, a DeepONet-style model, and CNO—across five PDE families spanning dispersive, elliptic, fluid, financial, and chaotic regimes: nonlinear Schrödinger, Poisson, Navier–Stokes, Black–Scholes, and Kuramoto–Sivashinsky. We then apply controlled shifts in parameters, boundary or terminal conditions, resolution, rollout horizon, and input perturbations. This setup lets us

ask a concrete ML question: does strong in-distribution performance predict robust operator learning, and if not, how do failures depend on architecture and equation class?

Our contributions are:

(1) We formulate robustness in neural PDE solving as a structured generalization problem and introduce a stress-testing framework for evaluating it.

(2) We instantiate the framework across FNO, DeepONet-style, and CNO models on five qualitatively different PDE families under a common protocol.

(3) We combine degradation-based metrics with spectral and rollout diagnostics to identify where robustness breaks down.

(4) We show that robustness is not captured by baseline accuracy alone and does not transfer cleanly across equations or stress conditions.

Overall, the paper argues that robustness claims for neural PDE solvers should be grounded in explicit evaluation under structured shift, not inferred from in-distribution accuracy alone.

2 Related Work

Neural operator methods learn mappings between function spaces rather than solution fields tied to a single discretization. Foundational work in this direction includes the general neural operator framework of Kovachki et al. [7], the Fourier Neural Operator (FNO) of Li et al. [14], and DeepONet [17], which established operator learning as a central paradigm for PDE surrogate modeling. More recent reviews synthesize the area from approximation-theoretic, algorithmic, and numerical-linear-algebra viewpoints, clarifying how architecture, discretization, data generation, and solver structure interact in operator learning [8].

The architecture space has also broadened substantially. Physics-Informed Neural Operators (PINO) incorporate PDE residual structure into operator learning [16]. Convolutional Neural Operators (CNOs) emphasize continuous–discrete consistency and robustness across resolutions and distributions [23]. FourCastNet illustrates the scale and practical importance of operator-style forecasting in complex spatiotemporal systems [21]. Beyond these, later variants target distinct inductive biases, including multiscale U-shaped designs, wavelet-localized representations, geometry-aware operators for irregular domains, and attention-based operator learners based on Fourier/Galerkin attention and transformer encoders [22; 29; 15; 1; 13]. Taken together, this literature suggests that robustness should be read through architecture-specific approximation bias rather than through a single notion of model quality.

Recent theory has started to sharpen those biases. For FNOs, existing work studies capacity, generalization, expressivity, trainability, truncation, and discretization effects, including Rademacher bounds, mean-field analysis, and decompositions of statistical, truncation, and grid-induced error [3; 6; 26; 11]. Related variants explicitly target settings where standard spectral parameterizations are stressed, including highly oscillatory operators and hyperbolic conservation laws [31; 4]. For DeepONet, recent work analyzes optimization, generalization, and required model size, while multiscale variants aim to mitigate failure on high-frequency operators [12; 19; 30].

At the same time, understanding when scientific machine learning models fail remains an open problem. In the PINN literature, Krishnapriyan et al. [9] showed that failure can arise from optimization and conditioning difficulties rather than lack of expressivity. More broadly, spectral bias remains a persistent limitation when high-frequency or multiscale structure must be recovered [28]. In operator learning, discontinuous and interface-dominated problems expose sharper structural limits. In particular, methods with linear reconstruction can be fundamentally inefficient for PDEs with discontinuities, motivating nonlinear reconstruction and discontinuity-aware extensions [10]. More recent discontinuity-focused operator architectures further extend this line by embedding interface structure directly into the learned representation [24]. These results strengthen the broader scientific-ML concern that strong in-distribution accuracy need not imply robustness under qualitatively different solution structure [2].

A particularly important and still under-emphasized issue is conditional generalization under varying boundary or terminal conditions. Recent work has argued that, when such conditions vary across samples, neural PDE solvers are often better interpreted as learning boundary-indexed families of operators rather than a single boundary-agnostic solution map [25]. Complementary work proposes explicit mechanisms for conditioning neural operators on complex boundary data, including learned boundary-to-domain extensions, and shows that boundary sensitivity can dominate performance in elliptic settings [18]. At the evaluation level, broader benchmarking efforts have expanded the empirical landscape through large curated suites and rollout-centered protocols, including PDEBench, APEBench, and The Well [27; 5; 20]. What remains less developed is a unified evaluation framework that probes deployment-relevant failure modes across multiple PDE families, architectures, and stress conditions under a common protocol. Our work is aimed at that gap: we develop a systematic stress-testing framework for comparing the robustness of neural PDE solvers under controlled shifts in parameters, boundary or terminal conditions, resolution, rollout horizon, and input perturbations.

3 Methodology

3.1 Architectures and Evaluation Design

We study robustness in neural PDE solvers through a standardized stress-testing framework. We evaluate three representative architectures under a common protocol: the Fourier Neural Operator (FNO) [14], DeepONet [17], and a convolutional neural operator (CNO) [23]. These models represent distinct inductive biases: spectral convolution in FNO, branch–trunk operator approximation in DeepONet, and localized multi-scale convolution in CNO.

For each PDE family, we generate paired input–output data from an in-distribution regime. Inputs include the relevant forcing terms, coefficients, initial conditions, payoff functions, or parameter channels, depending on the problem, and outputs are either static solution fields or short-horizon targets for time-dependent systems. For PDEs with known scalar parameters, such as viscosity ν in Navier–Stokes, nonlinearity κ in NLS, or volatility σ in Black–Scholes, the parameter is included as an input channel.

The goal is robustness profiling under a fixed evaluation protocol, not a final architecture leaderboard. We therefore use standardized implementations, a shared data-generation pipeline within each PDE family, and broadly comparable model scales, while allowing minor architecture-specific choices needed for stable optimization.

3.2 Stress Tests

After training on the in-distribution regime, each model is evaluated under five deployment-relevant shifts:

(A) **Parameter or coefficient shift:** parameters are moved beyond the training range, such as larger κ in NLS, lower viscosity ν in Navier–Stokes, higher volatility σ in Black–Scholes, or rougher coefficients in Poisson.

(B) **Boundary or terminal-condition shift:** models are tested on unseen boundary or payoff families.

(C) **Resolution extrapolation:** models trained at one discretization are evaluated at finer or coarser resolutions, with spectral diagnostics used to localize error across frequencies.

(D) **Long-horizon rollout:** one-step predictors are iterated beyond the training horizon to measure error accumulation and dynamical instability.

(E) **Input perturbation sensitivity:** small perturbations are added to state channels to test local stability under corrupted inputs.

Not every stressor applies to every PDE family, but within each PDE the stress grid is fixed and shared across architectures. All evaluations are performed without fine-tuning on the shifted regime.

3.3 Metrics and Multi-Seed Aggregation

We use a deterministic multi-seed protocol. For each architecture–PDE pair, we train 50 independent models, yielding 750 trained models in total. Every trained model is evaluated on the same stress suite.

Our primary summary statistic is the *degradation factor*,

$$D^{(i)} = \frac{E^{(i)}_{\text{stress}}}{E^{(i)}_{\text{base}}},$$

where $E^{(i)}_{\text{base}}$ is the baseline relative L^2 error of model i on in-distribution test data and $E^{(i)}_{\text{stress}}$ is the worst-case error over the corresponding stress grid. This measures robustness relative to baseline accuracy rather than in absolute terms.

We do not rely on degradation alone. For each architecture and PDE family, we also report absolute baseline L^2 error, rollout growth rate, rollout amplification, frequency-binned spectral error summaries under resolution shift, and 95% confidence intervals across seeds. This allows robustness to be assessed through multiple complementary diagnostics rather than a single summary score.

3.4 Experimental Setup

Training and evaluation are deterministic for each seed. For Poisson, we train on 512 samples at resolution $n = 128$ with batch size 8 for 3000 steps. For Black–Scholes, we use 512 samples at $n = 256$, batch size 8, and 3000 steps. For nonlinear Schrödinger, we use 256 samples at $n = 256$ and $n_t = 20$, batch size 4, and 4000 steps. For Navier–Stokes, we use 128 samples at $n = 64$ and $n_t = 20$, batch size 2, and 5000 steps. For Kuramoto–Sivashinsky, we use 256 samples at $n = 128$ and $n_t = 20$, batch size 8, and 3000 steps. All models are trained with Adam at learning rate 10^{-3}.

Model sizes are architecture-appropriate but broadly comparable. FNO uses width 64 and depth 4, with 16 Fourier modes in 1D and 12×12 modes in 2D. CNO uses width 64 and depth 5. The DeepONet-style model uses width 128 and depth 2. All models use coordinate channels. Baseline evaluation uses 64 in-distribution test samples per seed; for time-dependent PDEs, baseline rollout summaries use 5 rollout steps. Raw seed-level outputs are then aggregated into per-PDE summaries, cross-PDE comparison tables, and paper figures using the same analysis pipeline.

4 Results

Robustness is not predicted by baseline accuracy alone. Although FNO attains the lowest in-distribution error across all five PDE families, that advantage is not stable under structured shift. Under changes in parameters, boundary or terminal conditions, resolution, rollout horizon, and input perturbations, robustness depends jointly on architecture and equation class, and model rankings frequently change.

Figure 1 makes two points clear. First, baseline rankings do not predict robustness under shift. Second, the dominant failure mode is PDE-dependent: Poisson is driven by perturbation and resolution sensitivity, Navier–Stokes by rollout instability, Black–Scholes by payoff-family shift, and nonlinear Schrödinger by parameter extrapolation. These conclusions are supported not only by degradation factors, but also by the accompanying spectral and rollout diagnostics.

4.1 Nonlinear Schrödinger and Poisson

For nonlinear Schrödinger, parameter shift in κ is a shared difficulty across all three architectures, but resolution transfer separates them sharply. FNO and DeepONet remain essentially unchanged, while CNO degrades substantially. This contrast shows that some weaknesses are intrinsic to the problem, while others depend strongly on model design.

Poisson provides the most striking reversal. FNO is clearly best at baseline, yet becomes the least robust model under perturbation and resolution shift. CNO shows the opposite behavior, remaining stable across

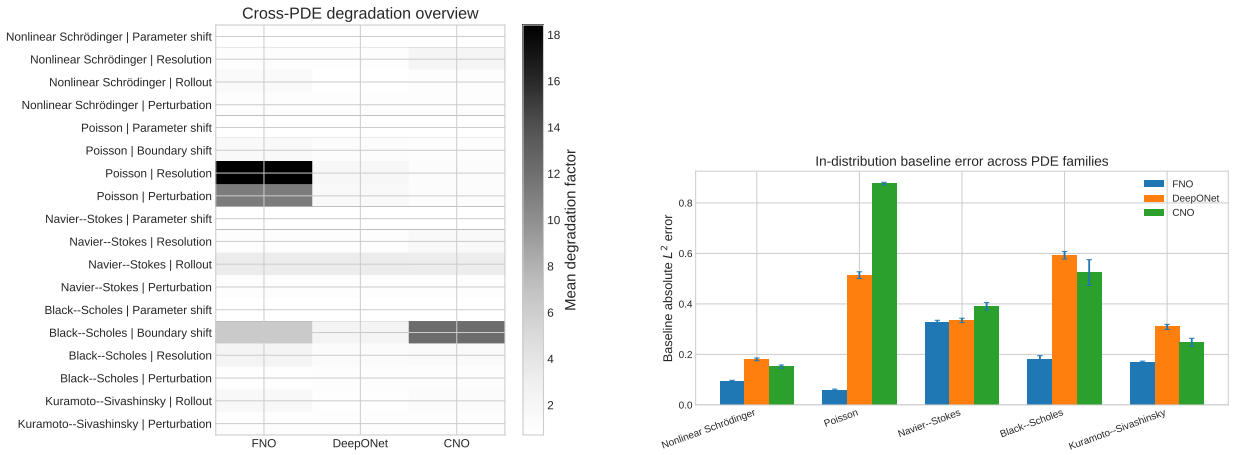


Figure 1: Left: mean degradation factors across PDE families, architectures, and stress tests. Darker cells indicate more severe degradation. Right: in-distribution baseline absolute L^2 error. FNO is strongest at baseline, but robustness patterns are far less uniform.

all stressors despite the worst baseline error. This inversion would be invisible under standard in-distribution evaluation and illustrates how model rankings depend on the intended deployment regime.

4.2 Navier–Stokes and Black–Scholes

Navier–Stokes does not produce a strong ranking inversion. Instead, it reveals a shared limitation: rollout instability. Parameter and perturbation shifts are mild for all models, and resolution transfer only moderately separates them. However, all three degrade substantially under long-horizon rollout. Here the main takeaway is not which model is best, but that current architectures struggle with iterative prediction regardless of baseline performance.

Black–Scholes exhibits a different pattern. The primary challenge is conditional generalization under payoff shift. DeepONet is consistently the most robust model across volatility, payoff, and resolution changes, despite having the worst baseline error. CNO, in contrast, fails severely under payoff shift. This indicates that different PDE families probe different generalization mechanisms: dynamical stability for Navier–Stokes versus functional extrapolation for Black–Scholes.

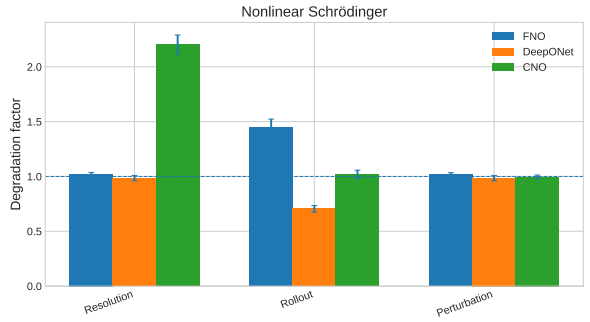
4.3 Kuramoto–Sivashinsky

Kuramoto–Sivashinsky isolates rollout behavior in a chaotic setting. FNO has the best baseline accuracy but the worst rollout degradation, while DeepONet shows the opposite pattern. This decoupling indicates that one-step accuracy and long-horizon stability are only weakly related, even in a simplified two-stressor setting.

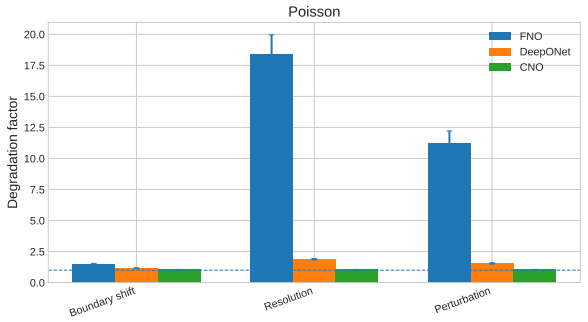
Takeaways

Three patterns emerge. First, baseline accuracy is not a reliable proxy for robustness. Second, some weaknesses are shared (notably rollout instability in Navier–Stokes), while others are highly architecture-dependent (e.g., Poisson resolution sensitivity or Black–Scholes payoff shift). Third, robustness varies significantly across PDE families: the stressor that matters most is problem-dependent.

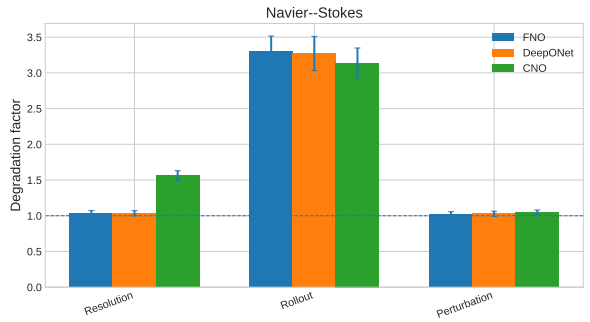
Taken together, these results show that robustness claims do not transfer cleanly across equations or shift types. Evaluating models along a single axis—whether baseline error, one PDE, or one perturbation—can give a misleading picture of performance under realistic deployment conditions.



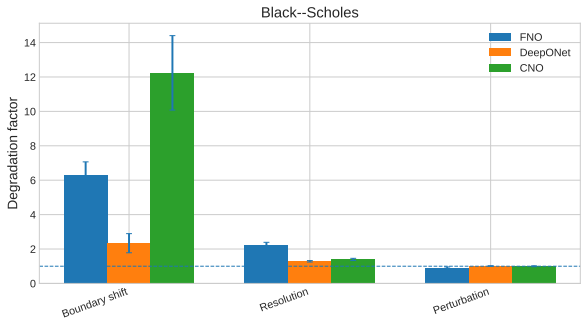
(a) Nonlinear Schrödinger. Parameter shift is broadly difficult, while resolution transfer is strongly architecture-dependent.



(b) Poisson. FNO is best at baseline but least robust under perturbation and resolution shift, whereas CNO is the most stable.



(c) Navier–Stokes. Rollout instability dominates and affects all architectures.



(d) Black–Scholes. DeepONet is most robust under payoff and volatility shift, while CNO exhibits severe degradation.

Figure 2: Degradation-factor summaries across four PDE families. The dominant failure mode differs substantially by equation class, and the strongest baseline model is not always the most robust under stress.

5 Discussion

The main lesson of this study is that robustness in neural PDE solvers is not a single property. It depends jointly on the PDE family, the stress type, and the architecture. Baseline accuracy therefore gives an incomplete picture: FNO is strongest in-distribution across all five PDE families, yet it is not uniformly the most robust under shift. DeepONet is often more stable under the most consequential shifts, while CNO shows the largest variance across settings. Model choice therefore depends not only on interpolation accuracy, but on which failure modes matter in deployment.

The observed failures are partly shared and partly architecture-specific. Navier–Stokes rollout degradation is the clearest shared limitation, with all three architectures deteriorating under long-horizon prediction. By contrast, Poisson shows a strong ranking reversal: FNO is best at baseline but least robust under perturbation and resolution shift, whereas CNO is the most stable. Black–Scholes reveals a different weakness centered on payoff-family generalization rather than rollout, and there DeepONet is the most robust model. Robustness is thus better understood as a structured profile than as a single score.

These results also clarify the scope of the study. The five PDE families span qualitatively different regimes, but they are still only a small sample of the PDEs practitioners may care about. The goal is not universal coverage, but a reusable evaluation template. In that sense, the contribution is not only the benchmark reported here, but also a standardized framework for comparing robustness under structured shift.

For practitioners, the implication is simple: the model with the lowest standard test error may not be the safest model once the operating regime changes. For researchers, the results point to concrete directions for

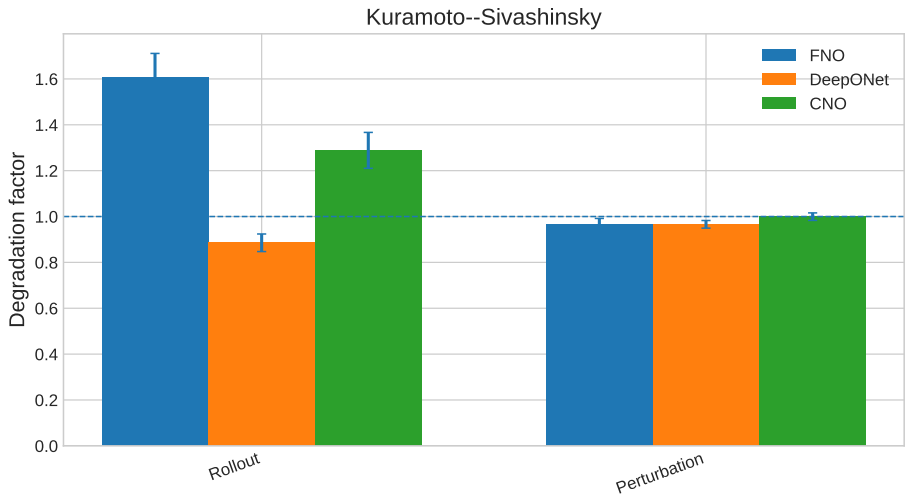


Figure 3: Kuramoto–Sivashinsky. The best baseline model and the most stable rollout model are different.

improvement, including stronger multi-resolution representations, better handling of boundary or terminal-condition variation, and training schemes that improve long-horizon stability. More broadly, claims about robust neural PDE solving should be tied to explicit stress conditions and equation families rather than to in-distribution accuracy alone.

6 Conclusion

We presented a comparative stress-testing evaluation of neural PDE solvers across five PDE families, three architectures, and multiple deployment-relevant shifts. The central result is that robustness is not captured by baseline accuracy alone. Although FNO achieves the lowest in-distribution error across all five PDE families, robustness rankings change substantially under shift. DeepONet is often the most consistently stable model, while CNO shows the greatest variability across equation classes.

More broadly, robustness depends on both the PDE family and the stress condition. Some weaknesses are shared, such as rollout instability in Navier–Stokes, while others are highly architecture-specific, such as Poisson sensitivity to perturbation and resolution or Black–Scholes sensitivity to payoff-family shift. No single benchmark, PDE, or perturbation is therefore sufficient for assessing reliability in neural PDE solvers.

The broader contribution of the paper is the evaluation framework itself. The same stress-based protocol can be extended to other PDE collections, architectures, and application-specific regimes, making it useful beyond the particular benchmark reported here. Taken together, the results suggest that robustness claims for neural PDE solvers should be grounded in explicit evaluation under structured shift rather than inferred from in-distribution accuracy alone.

Acknowledgements

Reproduciblity. Code to reproduce all experiments, generate figures, and compute degradation metrics is available at `REDACTED`.

Computational Resources. The authors gratefully acknowledge the provision of computational resources that supported the experiments in this work. All experiments were conducted on a high-performance workstation equipped with modern CPU and GPU hardware. The views and conclusions expressed herein are those of the authors and do not necessarily reflect the views of the supporting organization.

References

[1] Shuhao Cao. Choose a transformer: Fourier or galerkin, 2021. URL https://arxiv.org/abs/2105.14995.

[2] George Karniadakis, Yannis Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, pp. 1–19, 05 2021. doi: 10.1038/s42254-021-00314-5.

[3] Taeyoung Kim and Myungjoo Kang. Bounding the rademacher complexity of fourier neural operators, 2022. URL https://arxiv.org/abs/2209.05150.

[4] Taeyoung Kim and Myungjoo Kang. Approximating numerical fluxes using fourier neural operators for hyperbolic conservation laws, 2024. URL https://arxiv.org/abs/2401.01783.

[5] Felix Koehler, Simon Niedermayr, Rüdiger Westermann, and Nils Thuerey. Apebench: A benchmark for autoregressive neural emulators of pdes, 2024. URL https://arxiv.org/abs/2411.00180.

[6] Takeshi Koshizuka, Masahiro Fujisawa, Yusuke Tanaka, and Issei Sato. Understanding the expressivity and trainability of fourier neural operator: A mean-field perspective, 2024. URL https://arxiv.org/abs/2310.06379.

[7] Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces. *arXiv preprint arXiv:2108.08481*, 2021.

[8] Nikola B. Kovachki, Samuel Lanthaler, and Andrew M. Stuart. Operator learning: Algorithms and analysis, 2024. URL https://arxiv.org/abs/2402.15715.

[9] Aditi S. Krishnapriyan, Amir Gholami, Shandian Zhe, Robert M. Kirby, and Michael W. Mahoney. Characterizing possible failure modes in physics-informed neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 34, 2021. doi: 10.48550/arXiv.2109.01050. URL https://arxiv.org/abs/2109.01050.

[10] Samuel Lanthaler, Roberto Molinaro, Patrik Hadorn, and Siddhartha Mishra. Nonlinear reconstruction for operator learning of pdes with discontinuities, 2022. URL https://arxiv.org/abs/2210.01074.

[11] Samuel Lanthaler, Andrew M. Stuart, and Margaret Trautner. Discretization error of fourier neural operators, 2025. URL https://arxiv.org/abs/2405.02221.

[12] Sanghyun Lee and Yeonjong Shin. On the training and generalization of deep operator networks, 2023. URL https://arxiv.org/abs/2309.01020.

[13] Zijie Li, Kazem Meidani, and Amir Barati Farimani. Transformer for partial differential equations' operator learning, 2023. URL https://arxiv.org/abs/2205.13671.

[14] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *International Conference on Learning Representations (ICLR)*, 2021.

[15] 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, 2023. URL https://arxiv.org/abs/2309.00583.

[16] Zongyi Li, Hongkai Zheng, Nikola Kovachki, David Jin, Haoxuan Chen, Burigede Liu, Kamyar Azizzadenesheli, and Anima Anandkumar. Physics-informed neural operator for learning partial differential equations, 2023. URL https://arxiv.org/abs/2111.03794.

[17] 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.

[18] Sepehr Mousavi, Siddhartha Mishra, and Laura De Lorenzis. Imposing boundary conditions on neural operators via learned function extensions, 2026. URL https://arxiv.org/abs/2602.04923.

[19] Anirbit Mukherjee and Amartya Roy. Size lowerbounds for deep operator networks, 2024. URL https://arxiv.org/abs/2308.06338.

[20] Ruben Ohana, Michael McCabe, Lucas Meyer, Rudy Morel, Fruzsina J. Agocs, Miguel Beneitez, Marsha Berger, Blakesley Burkhart, Keaton Burns, Stuart B. Dalziel, Drummond B. Fielding, Daniel Fortunato, Jared A. Goldberg, Keiya Hirashima, Yan-Fei Jiang, Rich R. Kerswell, Suryanarayana Maddu, Jonah Miller, Payel Mukhopadhyay, Stefan S. Nixon, Jeff Shen, Romain Watteaux, Bruno Régaldo-Saint Blancard, François Rozet, Liam H. Parker, Miles Cranmer, and Shirley Ho. The well: a large-scale collection of diverse physics simulations for machine learning, 2025. URL https://arxiv.org/abs/2412.00568.

[21] Jaideep Pathak, Shashank Subramanian, Peter Harrington, Sanjeev Raja, Ashesh Chattopadhyay, Morteza Mardani, Thorsten Kurth, David Hall, Zongyi Li, Kamyar Azizzadenesheli, Pedram Hassanzadeh, Karth Kashinath, and Anima Anandkumar. Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators. *arXiv preprint arXiv:2202.11214*, 2022.

[22] Md Ashiqur Rahman, Zachary E. Ross, and Kamyar Azizzadenesheli. U-no: U-shaped neural operators, 2023. URL https://arxiv.org/abs/2204.11127.

[23] Bogdan Raonić, Roberto Molinaro, Tim De Ryck, Tobias Rohner, Francesca Bartolucci, Rima Alaifari, Siddhartha Mishra, and Emmanuel de Bézenac. Convolutional neural operators for robust and accurate learning of pdes, 2023. URL https://arxiv.org/abs/2302.01178.

[24] Sumanta Roy, Stephen T. Castonguay, Pratanu Roy, and Michael D. Shields. $\phi-$deeponet: A discontinuity capturing neural operator, 2026. URL https://arxiv.org/abs/2604.08076.

[25] Lennon Shikhman. One operator to rule them all? on boundary-indexed operator families in neural PDE solvers. In *AI&PDE: ICLR 2026 Workshop on AI and Partial Differential Equations*, 2026. URL https://openreview.net/forum?id=lDjWQ9UxRy.

[26] Unique Subedi and Ambuj Tewari. Controlling statistical, discretization, and truncation errors in learning fourier linear operators, 2025. URL https://arxiv.org/abs/2408.09004.

[27] Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Dan MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. PDEBench: An extensive benchmark for scientific machine learning. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. URL https://openreview.net/forum?id=dh_MkX0QfrK.

[28] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, and Jonathan T. Barron. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:7537–7547, 2020.

[29] Tapas Tripura and Souvik Chakraborty. Wavelet neural operator: a neural operator for parametric partial differential equations, 2022. URL https://arxiv.org/abs/2205.02191.

[30] Bo Wang, Lizuo Liu, and Wei Cai. Multi-scale deeponet (mscale-deeponet) for mitigating spectral bias in learning high frequency operators of oscillatory functions, 2025. URL https://arxiv.org/abs/2504.10932.

[31] Zhilin You, Zhenli Xu, and Wei Cai. Mscalefno: Multi-scale fourier neural operator learning for oscillatory function spaces, 2024. URL https://arxiv.org/abs/2412.20183.

Appendices

A Detailed Degradation Statistics

This appendix reports quantitative summaries of the stress-test results underlying the figures in the main text. For each architecture–PDE pair, we train 50 independent models and aggregate the resulting seed-level evaluations into summary statistics. We report the mean degradation factor, standard deviation, and 95% confidence interval across seeds, together with supporting diagnostics where relevant.

Unless otherwise noted, degradation factors are computed using the worst-case error over the fixed stress grid for a given scenario, for example the maximum error across parameter values, resolutions, or rollout horizons. This yields a conservative summary of failure severity under each stress type. Across runs, models differ only through initialization and the randomly generated training and evaluation samples drawn from the same in-distribution regime; the stress grids, aggregation rules, and analysis pipeline are otherwise held fixed.

A.1 Poisson Equation

Table 1: Poisson degradation summary (50 runs per architecture).

Architecture	Stress test	Mean	Std	$CI_{95\%}$ (low)	$CI_{95\%}$ (high)
FNO	Parameter shift (a scale)	1.994	0.449	1.870	2.118
	Boundary shift	1.498	0.194	1.444	1.551
	Resolution extrapolation	18.417	5.572	16.872	19.961
	Input perturbation	11.199	3.682	10.178	12.219
DeepONet	Parameter shift (a scale)	1.265	0.286	1.186	1.344
	Boundary shift	1.155	0.129	1.119	1.190
	Resolution extrapolation	1.882	0.195	1.828	1.936
	Input perturbation	1.545	0.184	1.494	1.596
CNO	Parameter shift (a scale)	1.087	0.136	1.049	1.124
	Boundary shift	1.049	0.035	1.039	1.058
	Resolution extrapolation	1.051	0.018	1.046	1.056
	Input perturbation	1.055	0.024	1.048	1.062

A.2 Nonlinear Schrödinger Equation

Table 2: Nonlinear Schrödinger degradation summary (50 runs per architecture).

Architecture	Stress test	Mean	Std	$CI_{95\%}$ (low)	$CI_{95\%}$ (high)
FNO	Nonlinearity shift (κ)	3.872	0.961	3.605	4.138
	Resolution extrapolation	1.022	0.053	1.008	1.037
	Long-horizon rollout	1.449	0.265	1.376	1.523
	Input perturbation	1.020	0.052	1.005	1.034
DeepONet	Nonlinearity shift (κ)	2.227	0.503	2.087	2.366
	Resolution extrapolation	0.985	0.088	0.960	1.009
	Long-horizon rollout	0.705	0.108	0.675	0.735
	Input perturbation	0.985	0.088	0.960	1.009
CNO	Nonlinearity shift (κ)	2.618	0.610	2.448	2.787
	Resolution extrapolation	2.203	0.315	2.115	2.290
	Long-horizon rollout	1.020	0.136	0.982	1.057
	Input perturbation	0.995	0.064	0.978	1.013

A.3 Navier–Stokes Equation

Table 3: Navier–Stokes degradation summary (50 runs per architecture).

Architecture	Stress test	Mean	Std	$CI_{95\%}$ (low)	$CI_{95\%}$ (high)
FNO	Viscosity shift (ν)	1.028	0.141	0.989	1.067
	Resolution extrapolation	1.034	0.138	0.996	1.072
	Long-horizon rollout	3.296	0.792	3.077	3.516
	Input perturbation	1.018	0.139	0.979	1.057
DeepONet	Viscosity shift (ν)	1.036	0.131	1.000	1.073
	Resolution extrapolation	1.035	0.130	0.999	1.071
	Long-horizon rollout	3.271	0.866	3.031	3.511
	Input perturbation	1.027	0.131	0.991	1.063
CNO	Viscosity shift (ν)	1.056	0.111	1.026	1.087
	Resolution extrapolation	1.562	0.248	1.493	1.630
	Long-horizon rollout	3.138	0.759	2.928	3.348
	Input perturbation	1.050	0.111	1.019	1.081

A.4 Black–Scholes Equation

Table 4: Black–Scholes degradation summary (50 runs per architecture).

Architecture	Stress test	Mean	Std	$CI_{95\%}$ (low)	$CI_{95\%}$ (high)
FNO	Volatility shift (σ)	2.068	0.656	1.886	2.250
	Payoff structure shift	6.308	2.733	5.550	7.065
	Resolution extrapolation	2.236	0.552	2.083	2.389
	Input perturbation	0.912	0.170	0.864	0.959
DeepONet	Volatility shift (σ)	1.272	0.118	1.239	1.304
	Payoff structure shift	2.338	2.005	1.782	2.894
	Resolution extrapolation	1.287	0.154	1.244	1.330
	Input perturbation	1.012	0.059	0.995	1.028
CNO	Volatility shift (σ)	1.357	0.203	1.301	1.413
	Payoff structure shift	12.239	7.805	10.076	14.403
	Resolution extrapolation	1.395	0.229	1.331	1.458
	Input perturbation	0.997	0.096	0.970	1.024

A.5 Kuramoto–Sivashinsky Equation

Table 5: Kuramoto–Sivashinsky degradation summary (50 runs per architecture).

Architecture	Stress test	Mean	Std	$CI_{95\%}$ (low)	$CI_{95\%}$ (high)
FNO	Long-horizon rollout	1.607	0.376	1.503	1.711
	Input perturbation	0.967	0.091	0.941	0.992
DeepONet	Long-horizon rollout	0.886	0.138	0.847	0.924
	Input perturbation	0.966	0.060	0.949	0.983
CNO	Long-horizon rollout	1.288	0.283	1.210	1.367
	Input perturbation	1.000	0.059	0.983	1.016

A.6 Implementation and Aggregation Details

All results are produced by a deterministic multi-seed evaluation pipeline. For each architecture–PDE pair, we run fixed seed sweeps with identical stress grids, shared data-generation settings within each PDE

family, and architecture-specific model templates held fixed across seeds. This ensures that differences across architectures reflect model behavior under the same evaluation protocol rather than variation in stress definitions or aggregation logic.

At the seed level, each run stores raw outputs for the baseline regime and for every stress test, including parameter or coefficient shifts, boundary or terminal-condition shifts, resolution extrapolation, rollout horizon shifts, and perturbation sensitivity. The perturbation protocol is applied to state channels by default, while spatially constant parameter channels are left unchanged, so the reported sensitivity reflects instability to corrupted states rather than trivial parameter noise. For time-dependent PDEs, rollout experiments store full error curves in addition to one-step errors.

Aggregation is performed separately from training and evaluation. For each architecture–PDE pair, we summarize seed-level results into paper-ready tables containing mean, standard deviation, and 95% confidence intervals. We report degradation factors together with auxiliary diagnostics including absolute baseline L^2 error, rollout growth rate, rollout amplification, and frequency-binned spectral error summaries. Cross-PDE comparison matrices and all paper figures are generated from these aggregated summaries rather than from ad hoc post-processing. This separation between raw evaluation, aggregation, and figure generation is intended to make the framework reusable and extensible for future robustness studies.