

Selecting PINNs and neural operators for repeated engineering analysis

Vladimir Shapovalov
The Polytechnic University of Turin

Abstract

Choosing a learned engineering surrogate requires specifying what is reused across physical cases, what information is available at prediction time, and which errors influence the design decision. Physics-informed neural networks and neural operators address overlapping but nonidentical versions of this task. This review compares coordinate-based field approximations, parameter-conditioned surrogates, function-to-function operators, and models refined with physical equations at deployment. A common mathematical description separates input sampling, learned approximation, output reconstruction, and case-specific adaptation. Analytical examples establish an input-information limit, a cost break-even condition, and a bound connecting surrogate-gradient error to design error under strong convexity. Applications to flow design, thermal loading, and viscoplastic response show why fast field prediction alone does not establish an engineering advantage. A proposed evaluation protocol compares physical accuracy, derivative accuracy, complete computational cost, and performance over genuinely unseen cases.

1. Start with the repeated engineering task

A conventional numerical solver maps a specified physical problem to an approximate solution. A PINN can represent that solution as a neural field trained using governing equations, boundary information, and observations [1, 2]. When additional parameters enter the network, the same model can represent multiple cases. Therefore, the distinction between PINNs and operators is not simply that one must always retrain and the other never does. The relevant distinction is the input family, its representation, and the information used during fitting and prediction.

Frequency-compensated flow prediction provides an example of conditioning on design and operating parameters [3]. Rate- and temperature-dependent plasticity provides a mechanically different parameterized setting [4]. In both cases, reuse is meaningful only over the represented problem family. A model for selected scalar parameters does not automatically accept arbitrary inlet profiles, spatial material distributions, or loading histories. Conversely, a functional input can be restricted to a low-dimensional family in a particular engineering application.

The related fluid-design patent describes a system in which learned frequency and phase information supports flow prediction for design candidates [5]. This is relevant to the workflow being considered, but the patent is not independent evidence of speed, accuracy, or successful optimization. Those claims require comparisons using the same design objectives and feasibility tolerances. This review treats architecture choice as a decision about a complete analysis workflow rather than a contest between model labels.

2. A common mathematical description

Let a denote the physical input, which may include a coefficient field, initial condition, boundary history, geometry description, and scalar parameters. Let G map this input to a solution field u . A practical learned predictor usually acts on finite observations $P_m a$ and reconstructs a field through an output representation R_n :

$$u = G(a), \hat{u} = R_n G_\theta(P_m a).$$

The finite input map P_m can sample an inlet profile at sensors or store a material field on a grid. The reconstruction R_n can interpolate grid outputs or evaluate a coordinate network. These maps belong in the model specification because they determine which distinctions between physical inputs survive preprocessing and which output quantities are available to an engineer. Increasing network width cannot recover information that the input map discarded.

A coordinate PINN for one case represents $u_\theta(x,t)$. A parameter-conditioned version represents $u_\theta(x,t;\mu)$, where μ may describe a finite-dimensional family. DeepONet combines a representation of sampled input functions with a representation of output coordinates [6]. A Fourier neural operator uses learned Fourier-space transformations in an architecture intended to approximate solution operators [7]. These constructions differ from adding Fourier features to coordinates, which changes a coordinate network's representation without itself specifying a function-to-function map [8].

Physics-informed operators combine operator learning with equation constraints [9, 10]. The equations may be used during training, target-case refinement, or both. Each use changes the information and computational budget. An operator that predicts once from a frozen model should not be compared as if it performed the same task as a model subsequently optimized on the target geometry. The latter may be valuable, but the additional work and physical information must be counted.

Model construction	Object represented	Information that must be declared
One-case coordinate PINN	A field over coordinates	Equations, observations, and case-specific training
Parameter-conditioned PINN	Fields over a parameter family	Parameter ranges and represented combinations
Neural operator	A mapping between input and output fields	Input sampling and output discretization
Physics-refined operator	An operator prediction followed by adaptation	Target equations, data access, and adaptation budget

3. An information limit before architecture selection

Suppose two physically distinct inputs a_1 and a_2 have exactly the same encoded representation, $P_m a_1 = P_m a_2$. A deterministic predictor using only that representation must return the same output v for both. For any output norm, the triangle inequality gives

$$\begin{aligned} \|G(a_1) - G(a_2)\| &\leq \|G(a_1) - v\| + \|G(a_2) - v\|. \\ \max(\|G(a_1) - v\|, \|G(a_2) - v\|) &\geq (1/2)\|G(a_1) - G(a_2)\|. \end{aligned}$$

At least one error must therefore be at least half the separation between the true outputs. This is an elementary information bound, independent of optimization and network capacity. It applies equally to sampled operator inputs and an inadequately parameterized coordinate surrogate. It does not assert that every practical sensor arrangement has such an ambiguity; it states what follows when the ambiguity exists.

For an engineering illustration, two inlet-temperature profiles can agree at the input sensors while differing between them. If the governing transport problem responds to those differences, no model receiving only the identical sampled values can return both correct solutions. Similarly, a plasticity input containing only the current strain and temperature can omit the history needed to distinguish internal states. Resolving either problem requires a richer input representation or additional assumptions about the admissible input family, not merely a larger network.

The operator approximation results motivating DeepONet concern suitable continuous operators and adequately represented inputs; they do not remove optimization or generalization error [6]. Resolution-transfer demonstrations in FNO concern the tested problem families [7]. Neither result should be read as a guarantee that arbitrary engineering inputs can be reconstructed from a fixed insufficient sensor set. This distinction is especially important when geometry or boundary topology changes between cases.

4. Representation, discretization, and physical quantities

For a chosen input reconstruction R_m , the prediction error can be separated conceptually into error caused by replacing a with $R_m P_m a$ and error in approximating the resulting solution. If G is Lipschitz on the relevant input set with constant L_G , then

$$\|G(a) - \hat{u}\| \leq L_G \|a - R_m P_m a\| + \|G(R_m P_m a) - \hat{u}\|.$$

The inequality is conditional on the stated norm and Lipschitz property. Near bifurcations, shocks, loss of stability, or changes of constitutive regime, a useful uniform constant may be unavailable. Even when it holds, the second term contains approximation, discretization, training, and evaluation errors. The decomposition organizes these potentially interacting error contributions.

Engineering outputs often differentiate the field. Heat flux depends on a temperature gradient, viscous traction on velocity gradients, and elastic stress on displacement derivatives. A small field error in an average norm is insufficient to guarantee those quantities. Mixed-output mechanical and flow formulations explicitly alter which variables are approximated and differentiated [11, 12,

13]. Thermal and additive-manufacturing applications likewise require validation of cooling rates, fluxes, and localized maxima [14, 15].

Fourier features, periodic activations, and multiscale frequency analyses provide routes to representing oscillatory structure [8, 16, 17]. They can also change derivative conditioning because differentiating a sinusoid introduces frequency factors. A fine output grid may display a smooth-looking high-resolution field without adding trustworthy physical frequencies. A resolution study should evaluate error against an independent converged reference, rather than treating the number of output pixels as an accuracy measure.

Irregular geometry introduces another representation choice. Graph-based mesh simulators operate on a discretization and its connectivity [18]. Their geometry-transfer properties depend on node attributes, connectivity, message passing, and the represented mesh family. A graph simulator is not automatically physics-informed in the residual-based sense. Geometry-aware architecture, supervised simulation learning, and explicit equation penalties are separate mechanisms and should be reported separately.

5. Cost accounting and an analytical break-even calculation

Let a conventional solver require cost C_s per new case. Let the learned workflow require a fixed setup cost C_0 , prediction cost C_q per case, and adaptation cost C_a per case. Setup includes data generation, preprocessing, architecture searches, unsuccessful runs, and final training. For N cases,

$$\begin{aligned} C_{\text{solver}}(N) &= N C_s. \\ C_{\text{learned}}(N) &= C_0 + N(C_q + C_a). \\ N &> C_0 / (C_s - C_q - C_a), \text{ if } C_s > C_q + C_a. \end{aligned}$$

If the denominator is nonpositive, no positive number of repeated cases recovers the setup cost under this simplified model. Hardware, parallelism, batch sizes, and requested output quantities must be comparable. If a conventional solver also has substantial reusable setup, include its fixed cost on the other side of the comparison. The expression provides a starting model for task-specific cost accounting.

As a hypothetical arithmetic example, take a learned setup cost of 600 compute-hours, a reference solve cost of two hours per case, and frozen prediction cost of 0.02 hours. With no adaptation, the learned cost becomes lower at 304 integer cases. If adaptation adds 0.5 hours per case, the crossover moves to 406 cases. If adaptation costs two hours, there is no crossover against the stated reference solver. These assumed numbers illustrate sensitivity to adaptation.

The useful denominator should also reflect failures. If an out-of-envelope case triggers a conventional solve, the expected online cost includes that fallback probability. Model maintenance, retraining after geometry changes, and independent verification add further costs. A surrogate that fails to meet the engineering tolerance should not receive credit merely because its raw query is cheap. Published FEM comparisons underline the importance of distinguishing trained-network evaluation from the full cost of obtaining an accurate solution [19].

Multifidelity training can reduce the cost of obtaining information, but introduces a relationship between low- and high-fidelity outputs that must itself be learned and checked [20]. A systematic bias in the inexpensive model does not vanish when its dataset becomes large. The training-data bill should include the number and accuracy of all reference solves, not only the smaller high-fidelity subset.

6. A design optimizer needs trustworthy derivatives

A differentiable surrogate can supply objective gradients to an optimizer, but differentiability is not the same as gradient accuracy. Let $J(\mu)$ be a differentiable true design objective that is m -strongly convex over a convex feasible domain, with $m > 0$. Suppose an interior surrogate optimum μ_h satisfies $\nabla J_h(\mu_h) = 0$ and the gradient discrepancy at that point obeys $\|\nabla J(\mu_h) - \nabla J_h(\mu_h)\| \leq \delta$. If the true optimum μ^* is interior, strong monotonicity of ∇J yields

$$\|\mu_h - \mu^*\| \leq \delta/m.$$

The argument uses $m\|\mu_h - \mu^*\|^2 \leq (\nabla J(\mu_h) - \nabla J(\mu^*)) \cdot (\mu_h - \mu^*)$ and $\nabla J(\mu^*) = 0$, followed by Cauchy-Schwarz. This is a local workflow diagnostic under restrictive assumptions, not a guarantee for nonconvex fluid-design optimization. Active constraints require an appropriate optimality-condition analysis, and multiple local optima can defeat a global-distance interpretation.

The example clarifies why reporting only solution-field error leaves a gap in a design claim. Validate sensitivities by independent perturbations or trusted reference derivatives. Then evaluate the proposed optimum with a conventional solver, including feasibility margins. Hard-constraint inverse-design methods provide a relevant formulation reference [21], while exact boundary parameterizations illustrate how selected constraints can be imposed structurally [22]. Neither eliminates surrogate model error in the objective or other constraints.

7. Flow and solid mechanics imply different reuse envelopes

For a flow surrogate, the relevant inputs may include geometry, inlet profile, viscosity, and time. A change in Reynolds number or geometry can alter wake frequency and phase, making a long-horizon prediction a different challenge from spatial interpolation. The frequency-compensated study reports improvements over a no-feature comparison for its selected conditions [3]. Its evidence should not be extended to untested turbulent regimes or surface loads outside the represented region. The fluid-design patent supplies application context without resolving that validation gap [5].

For viscoplasticity, loading path and internal variables matter alongside strain rate and temperature [4]. An operator intended to accept arbitrary load histories must represent the history or an adequate evolving state. A finite vector of operating parameters may be entirely sufficient for a deliberately restricted monotonic family. Thermodynamic constitutive networks provide a complementary route to structuring material response [23]. The correct choice depends on the admissible histories and the quantities used in the engineering decision.

This distinction also affects extrapolation tests. Holding out a scalar rate inside a represented range, extending the strain interval, and changing the loading path test different capabilities. A single pooled score hides those differences. Operator models should be tested on complete unseen functions or cases, while coordinate models should have a separately stated spatial verification grid. Both need tests that reflect the actual proposed reuse.

8. A proposed comparison that can support a decision

Specify a physical family, engineering tolerances, and anticipated number of cases before choosing models. Establish a converged conventional reference and a simple surrogate baseline. Then compare a parameter-conditioned PINN, a suitable operator, and a physics-refined variant only if the target workflow permits adaptation. Give each family an explicit tuning budget. Optimization pathology studies show why comparing a carefully tuned method with an unstable default configuration is unreliable [24, 25, 26]. Practical training guidance can help standardize the starting point [27].

Separate frozen prediction, physics-only adaptation, and adaptation with target measurements. Report field norms, derivative quantities, conservation diagnostics, design-ranking error, constraint misses, and total cost. Retain failed initializations. Residual sampling and temporal organization can materially affect training, so they must either be controlled or treated as separate interventions [28, 29]. A method should not gain an invisible advantage by using target-case physics or measurements that its baseline never receives.

PDEBench and PINNacle offer reusable benchmark contexts, and DeepXDE provides implementation infrastructure [30, 31, 32]. They do not replace application-specific tests of geometry, loading paths, and engineering quantities. Mathematical convergence results and inverse stability estimates apply under their stated assumptions [33, 34]. They are not automatic certification of a learned operator over a new physical family.

Bayesian PINNs and ensembles can provide uncertainty information under different assumptions [35, 36]. For a repeated design workflow, evaluate uncertainty on the final quantities and examine the cost and effectiveness of fallback decisions. The preferred model is the least costly complete workflow that meets the declared accuracy and feasibility requirements over the intended workload, with failures and maintenance included.

9. Worked input-resolution example for a conduction operator

Consider the nondimensional steady conduction problem $-u''(x) = a(x)$ on the unit interval with zero endpoint temperatures. Suppose the input encoder samples the source at $x_j = j/m$, for integer j from zero to m . Choose $a_1(x) = 0$ and $a_2(x) = \sin(2\pi mx)$. Every encoded source value is zero in both cases, yet the exact solutions are $u_1 = 0$ and $u_2 = \sin(2\pi mx)/(2\pi m)^2$. The two physical inputs are therefore indistinguishable to that encoder while their solutions differ.

For the maximum norm, any deterministic predictor based only on those samples has error at least $1/[2(2\pi m)^2]$ on one of the two cases. The corresponding heat-flux solutions, defined here as $-u'$, differ in maximum norm by $1/(2\pi m)$. If flux is the quantity of interest, the information bound is

consequently larger in its own nondimensional scale than the temperature bound. The example connects input aliasing directly to the engineering output rather than only to a visual reconstruction error.

The source family in this example changes with m . It therefore does not contradict convergence as sensors become denser for a fixed suitably regular input family. Instead, it shows why a claim of resolution independence must specify the admissible bandwidth or regularity of the inputs. Increasing output resolution without enriching the sampled source cannot distinguish the two stated cases. Additional PDE evaluations using the true source between input sensors would supply new information and change the problem being solved.

In an industrial thermal application, the analogue is a narrow heat-input feature missed by the supplied source grid. A physics-refined model may recover its effect if refinement has access to a sufficiently resolved source description. A frozen operator receiving only the aliased grid does not have that information. Comparisons should identify whether source reconstruction is performed outside the neural model and include its computational cost.

This small problem is also a useful implementation check. Its exact fields, input samples, and fluxes provide directly evaluable reference quantities. It isolates information loss from approximation and optimization errors. More realistic nonlinear experiments are still necessary, but they should preserve this distinction between a model that failed to use available information and a model that was never supplied enough information.

10. Conclusions

PINNs and neural operators should be compared through the physical mapping, information representation, and pattern of reuse. A coordinate surrogate can be reusable, an operator can require case-specific refinement, and either can be limited by inadequate input information. Fourier coordinate features and Fourier operator layers should be distinguished even when both help represent oscillatory behavior.

The analytical information bound, cost crossover, and derivative-error argument identify three separate engineering requirements: informative inputs, worthwhile amortization, and trustworthy downstream decisions. An architecture claim becomes useful when those requirements are tested together with physical accuracy and complete computational cost.

References

- [1] Maziar Raissi, Paris Perdikaris, George Em Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics* 378, 686-707 (2019). DOI.
- [2] George Em Karniadakis, Ioannis G. Kevrekidis, Lu Lu, et al.. Physics-informed machine learning. *Nature Reviews Physics* 3, 422-440 (2021). DOI.
- [3] Tongtao Zhang, Biswadip Dey, Pratik Kakkar, Arindam Dasgupta, Amit Chakraborty. Frequency-compensated PINNs for Fluid-dynamic Design Problems. *ML4Eng Workshop, NeurIPS 2020*. arXiv:2011.01456 (2020). arXiv.

- [4] Rajat Arora, Pratik Kakkar, Amit Chakraborty, Biswadip Dey. Physics-informed neural networks for modeling rate- and temperature-dependent plasticity. Machine Learning and the Physical Sciences Workshop, NeurIPS 2022. arXiv:2201.08363v3 (2022). arXiv.
- [5] Biswadip Dey, Pratik Kakkar, Ankur Kaushik, Arindam Dasgupta, Amit Chakraborty, Tongtao Zhang. Frequency-compensated neural networks for fluid dynamics design problems. WO2021163309A1. PCT/US2021/017635. Published 19 August 2021 (2021). Source.
- [6] Lu Lu, Pengzhan Jin, George Em Karniadakis. DeepONet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. arXiv:1910.03193 (2019). arXiv.
- [7] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, et al.. Fourier Neural Operator for Parametric Partial Differential Equations. arXiv:2010.08895 (2020). arXiv.
- [8] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, et al.. Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. arXiv:2006.10739 (2020). arXiv.
- [9] Zongyi Li, Hongkai Zheng, Nikola Kovachki, et al.. Physics-Informed Neural Operator for Learning Partial Differential Equations. arXiv:2111.03794 (2021). arXiv.
- [10] Sifan Wang, Hanwen Wang, Paris Perdikaris. Learning the solution operator of parametric partial differential equations with physics-informed DeepOnets. arXiv:2103.10974 (2021). arXiv.
- [11] Ehsan Haghighat, Maziar Raissi, Adrian Moure, et al.. A deep learning framework for solution and discovery in solid mechanics. arXiv:2003.02751 (2020). arXiv.
- [12] Chengping Rao, Hao Sun, Yang Liu. Physics-informed deep learning for incompressible laminar flows. arXiv:2002.10558 (2020). arXiv.
- [13] R. Sharma, Y. B. Guo. Thermal-Mechanical Physics Informed Deep Learning For Fast Prediction of Thermal Stress Evolution in Laser Metal Deposition. arXiv:2412.18786 (2024). arXiv.
- [14] Shuheng Liao, Tianju Xue, Jihoon Jeong, et al.. Hybrid thermal modeling of additive manufacturing processes using physics-informed neural networks for temperature prediction and parameter identification. arXiv:2206.07756 (2022). arXiv.
- [15] Qiming Zhu, Zeliang Liu, Jinhui Yan. Machine learning for metal additive manufacturing: Predicting temperature and melt pool fluid dynamics using physics-informed neural networks. arXiv:2008.13547 (2020). arXiv.
- [16] Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, et al.. Implicit Neural Representations with Periodic Activation Functions. arXiv:2006.09661 (2020). arXiv.
- [17] Sifan Wang, Hanwen Wang, Paris Perdikaris. On the eigenvector bias of Fourier feature networks: From regression to solving multi-scale PDEs with physics-informed neural networks. arXiv:2012.10047 (2020). arXiv.
- [18] Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, Peter W. Battaglia. Learning Mesh-Based Simulation with Graph Networks. arXiv:2010.03409 (2020). arXiv.
- [19] Tamara G. Grossmann, Urszula Julia Komorowska, Jonas Latz, Carola-Bibiane Schönlieb. Can Physics-Informed Neural Networks beat the Finite Element Method?. arXiv:2302.04107 (2023). arXiv.
- [20] Xuhui Meng, George Em Karniadakis. A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse PDE problems. arXiv:1903.00104 (2019). arXiv.
- [21] Lu Lu, Raphael Pestourie, Wenjie Yao, et al.. Physics-informed neural networks with hard constraints for inverse design. arXiv:2102.04626 (2021). arXiv.
- [22] N. Sukumar, Ankit Srivastava. Exact imposition of boundary conditions with distance functions in physics-informed deep neural networks. arXiv:2104.08426 (2021). arXiv.
- [23] Filippo Masi, Ioannis Stefanou, Paolo Vannucci, Victor Maffi-Berthier. Thermodynamics-based Artificial Neural Networks for constitutive modeling. arXiv:2005.12183 (2020). arXiv.
- [24] Sifan Wang, Yujun Teng, Paris Perdikaris. Understanding and mitigating gradient pathologies in physics-informed neural networks. arXiv:2001.04536 (2020). arXiv.

- [25] Sifan Wang, Xinling Yu, Paris Perdikaris. When and why PINNs fail to train: A neural tangent kernel perspective. arXiv:2007.14527 (2020). arXiv.
- [26] Aditi S. Krishnapriyan, Amir Gholami, Shandian Zhe, et al.. Characterizing possible failure modes in physics-informed neural networks. arXiv:2109.01050 (2021). arXiv.
- [27] Sifan Wang, Shyam Sankaran, Hanwen Wang, Paris Perdikaris. An Expert's Guide to Training Physics-informed Neural Networks. arXiv:2308.08468 (2023). arXiv.
- [28] Chenxi Wu, Min Zhu, Qinyang Tan, et al.. A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks. arXiv:2207.10289 (2022). arXiv.
- [29] Sifan Wang, Shyam Sankaran, Paris Perdikaris. Respecting causality is all you need for training physics-informed neural networks. arXiv:2203.07404 (2022). arXiv.
- [30] Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, et al.. PDEBENCH: An Extensive Benchmark for Scientific Machine Learning. arXiv:2210.07182 (2022). arXiv.
- [31] Zhongkai Hao, Jiachen Yao, Chang Su, et al.. PINNacle: A Comprehensive Benchmark of Physics-Informed Neural Networks for Solving PDEs. arXiv:2306.08827 (2023). arXiv.
- [32] Lu Lu, Xuhui Meng, Zhiping Mao, George E. Karniadakis. DeepXDE: A deep learning library for solving differential equations. arXiv:1907.04502 (2019). arXiv.
- [33] Yeonjong Shin, Jerome Darbon, George Em Karniadakis. On the convergence of physics informed neural networks for linear second-order elliptic and parabolic type PDEs. arXiv:2004.01806 (2020). arXiv.
- [34] Siddhartha Mishra, Roberto Molinaro. Estimates on the generalization error of Physics Informed Neural Networks (PINNs) for approximating a class of inverse problems for PDEs. arXiv:2007.01138 (2020). arXiv.
- [35] Liu Yang, Xuhui Meng, George Em Karniadakis. B-PINNs: Bayesian Physics-Informed Neural Networks for Forward and Inverse PDE Problems with Noisy Data. arXiv:2003.06097 (2020). arXiv.
- [36] Balaji Lakshminarayanan, Alexander Pritzel, Charles Blundell. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. arXiv:1612.01474 (2016). arXiv.