Journal of Interdisciplinary Engineering, Computing and Automation
Vol. 01, Issue 01, July–December 2026, pp. 49-60

Learning the Mesh Instead of the Physics: Discretisation Bias in Machine-Learning Surrogates for Engineering Simulation

Proshanta Sarkar

Assistant Professor, Department of Computer Science and Engineering, B. P. Poddar Institute of Management and Technology, Kolkata 700052, West Bengal, India.

Email: psarkar1710@gmail.com

ABSTRACT

Machine-learning surrogates for engineering simulation are trained on solver output and validated against held-out solver output. If the training simulations were run on an unconverged mesh, the surrogate reproduces the discretisation error faithfully and its reported accuracy remains excellent. We quantify the resulting bias, its interaction with training-set size, and a multi-fidelity remedy. We solved a one-dimensional variable-coefficient diffusion problem by linear finite elements over a three-parameter design space, treating a 4096-element solution as the converged reference. Polynomial surrogates were trained on meshes of 4 to 64 elements with 10 to 640 training points and evaluated twice: against held-out output from the same mesh, and against the reference. We then tested a discrepancy-correction scheme combining many coarse samples with a few converged ones. Solver error against the reference fell from 5.93e-3 at 4 elements to 2.32e-5 at 64, consistent with second-order convergence. Surrogates trained on the 4-element mesh reached an error of 3.27e-4 against their own mesh but 5.96e-3 against the reference, a factor of 18; on the 8-element mesh the factor was 4.4. Error against the reference stopped improving after 20 training points on the coarsest mesh and 80 on the next, so the remaining samples reduced variance around a biased target. On meshes of 16 elements and finer the discretisation error fell below the surrogate's own approximation error, and the binding floor changed accordingly. Adding 5 converged samples to 640 coarse ones reduced error from 1.53e-3 to 5.11e-4 at 1.13 times the solver cost. Held-out solver output cannot detect an error the solver itself contains. Surrogate accuracy should be reported against a convergence study, and a small budget of converged samples removes most of the inherited bias.

Keywords: surrogate modelling; finite element method; discretisation error; multi-fidelity; engineering informatics; model validation.

1. INTRODUCTION

Machine-learning surrogates are now routine in engineering design. A parametric finite element or computational fluid dynamics model is sampled across its design space, a regressor is fitted to the resulting input-output pairs, and the surrogate replaces the solver in optimisation, uncertainty quantification or real-time prediction [1-3,25-27]. The economics are compelling: a surrogate evaluates in microseconds where the solver takes minutes.

The surrogate is validated the way regression models are validated. Data are split, the model is fitted on one part and scored on the other, and a small held-out error is reported as evidence that the surrogate reproduces the simulation.

It is evidence of exactly that, and of nothing more. The held-out data came from the same solver on the same mesh, so any error the solver makes is present identically in training and test sets. A surrogate that has learned the solver's discretisation error perfectly will score perfectly, and its reported accuracy is silent about whether the underlying simulations were converged.

Mesh convergence is well understood as a solver matter and verification procedures for it are long established [4-6,16-18]. What changes when a surrogate is interposed is that the convergence question becomes easy to skip: training data generation is expensive, coarse meshes are the obvious economy, and the validation step that would normally catch the problem has been replaced by one that cannot [19-21].

We quantify the consequence on a problem where the converged answer is available, and ask three questions: how large is the gap between reported and actual accuracy, whether more training data closes it, and what a small budget of converged simulations buys.

1.1 Contributions

We measure the divergence between same-mesh and converged-reference validation error across mesh resolutions and training-set sizes; we identify the training-set size beyond which further data does not improve accuracy against the physics; we distinguish the discretisation floor from the surrogate's own capacity floor and show which binds where; and we quantify a discrepancy-correction scheme against a matched-cost single-fidelity baseline.

2. METHOD

2.1 Governing Problem

We use a one-dimensional variable-coefficient diffusion problem, chosen because it is a standard engineering model, because linear finite elements on it have well-characterised second-order convergence [14,15], and because a highly refined solution can serve as a reference:

$$-\frac{d}{dx}\left(k(x)\frac{du}{dx}\right)=f(x),u(0)=u(1)=0 \tag{1}$$

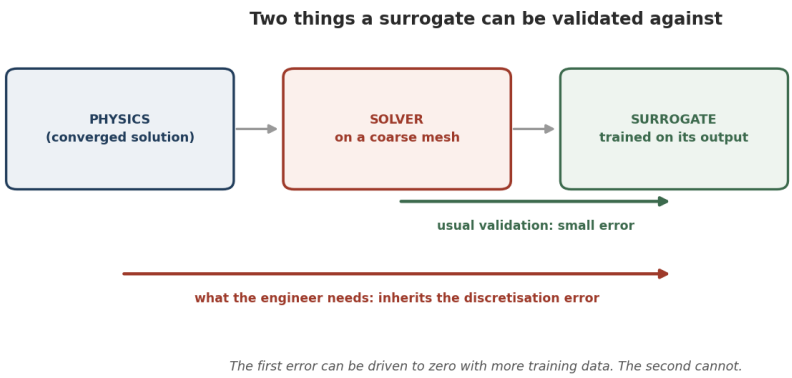


Fig.1: The two quantities a surrogate can be validated against.

The design space has three parameters. The conductivity is $k(x) = 1 + a \sin(\pi x) + bx$ and the source is $f(x) = 1 + c \cos(2\pi x)$, with a, b and c sampled uniformly over stated ranges. The quantity of interest is the mean of u over the domain.

Solutions use linear elements with one-point quadrature on the coefficients [18]. A 4096-element solution is treated as converged and used as the reference throughout; we verified that the solver exhibits the expected second-order convergence toward it before running any surrogate experiment. Figure 1 contrasts the two quantities a surrogate can be validated against.

2.2 Surrogate and Experiments

The surrogate is a total-degree 4 polynomial in the three parameters, fitted by ridge regression. A polynomial was chosen deliberately: its capacity is fixed and known, so its own approximation error can be separated from the discretisation error, which a neural network of unstated capacity would confound [19-24].

Training sets of 10 to 640 points were drawn for each of five meshes from 4 to 64 elements. Every surrogate was scored twice on a common test set of 600 parameter points: against the solver output on its own training mesh, which is the conventional validation, and against the reference solution.

A multi-fidelity variant was then tested. A surrogate fitted to a large coarse-mesh sample was corrected by a second polynomial fitted to the discrepancy between that surrogate and a small number of converged solves, and compared against single-fidelity surrogates trained only on converged solves at matched solver cost, counted in element-solves [8-11].

Parameters are given in Table 1. The work was implemented in Python using NumPy, SciPy and Matplotlib [28-30] and the complete script accompanies this paper.

Table 1: Experimental parameters.

Parameter	Value	Note
Governing problem	1D variable-coefficient diffusion	Standard model with known convergence behaviour
Design space	3 parameters, uniform	Conductivity shape and source shape
Quantity of interest	Domain mean of u	A scalar functional of the field
Reference mesh	4096 elements	Treated as converged
Training meshes	4, 8, 16, 32, 64 elements	Spans unconverged to near-converged
Training points	10 to 640	Logarithmic spacing
Surrogate	Degree-4 polynomial, ridge	Fixed, known capacity
Test set	600 points	Common to every configuration

3. RESULTS

3.1 The Solver Behaves As It Should

Figure 2 and Table 2 confirm that the solver converges. Error against the reference falls from 5.93e-3 at 4 elements to 2.32e-5 at 64, closely tracking a second-order slope. Expressed relative to the spread of the quantity of interest across the design space, the 4-element mesh carries an error of 25.0 per cent and the 64-element mesh 0.10 per cent.

This is unremarkable and it is the point. Nothing is wrong with the solver, and a conventional mesh convergence study would identify the coarse meshes as inadequate immediately [6]. The question is what happens when that study is not performed and a surrogate is fitted instead.

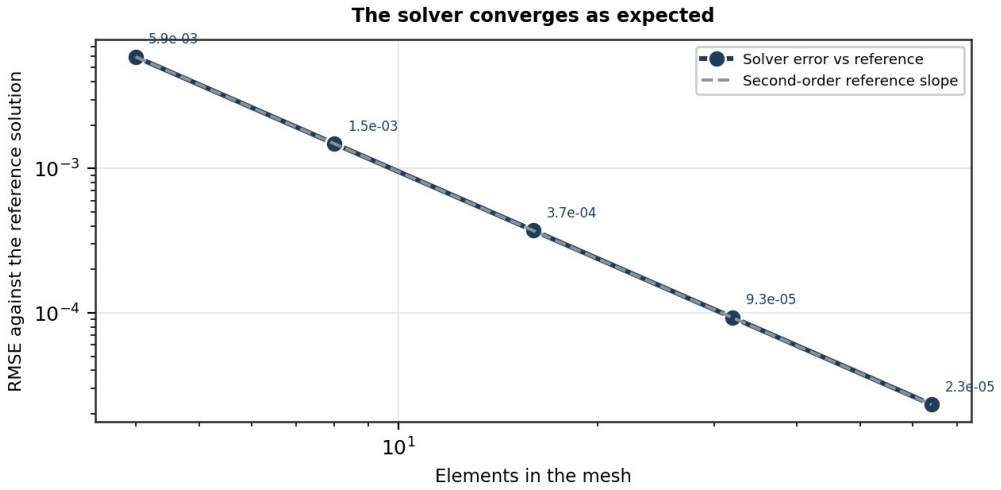


Fig.2: Solver error against the reference solution, with a second-order slope for comparison.

Table 2: Solver discretisation error.

Elements	RMSE vs reference	Bias	RMSE as % of QoI spread
4	5.93e-3	-5.37e-3	25.04%
8	1.48e-3	-1.36e-3	6.27%
16	3.71e-4	-3.40e-4	1.57%
32	9.29e-5	-8.51e-5	0.39%
64	2.32e-5	-2.13e-5	0.10%

3.2 Reported Accuracy and Actual Accuracy Diverge

Figure 3 shows learning curves for the two coarsest meshes, scored both ways. On the 4-element mesh the error against the training mesh falls steadily with more data, reaching 3.27e-4 at 640 points. The error against the reference falls for two steps and then stops, flattening at 5.96e-3 — a factor of 18 larger.

The flattening occurs exactly at the discretisation floor marked on the figure. Once the surrogate has learned the coarse-mesh solution to within its own accuracy, every further training point refines its agreement with a systematically wrong target.

Table 3 and Figure 4 give the gap at the largest training size for every mesh. It is a factor of 18 at 4 elements and 4.4 at 8, falling toward unity as the mesh refines and the discretisation error becomes negligible.

The practical reading is uncomfortable. A surrogate developer following standard practice on the 4-element mesh would report a held-out RMSE of 3.27e-4, a figure that is correct, reproducible and about 18 times better than the surrogate's actual agreement with the physics. No amount of care in the machine-learning workflow surfaces this, because the workflow never sees a converged solution.

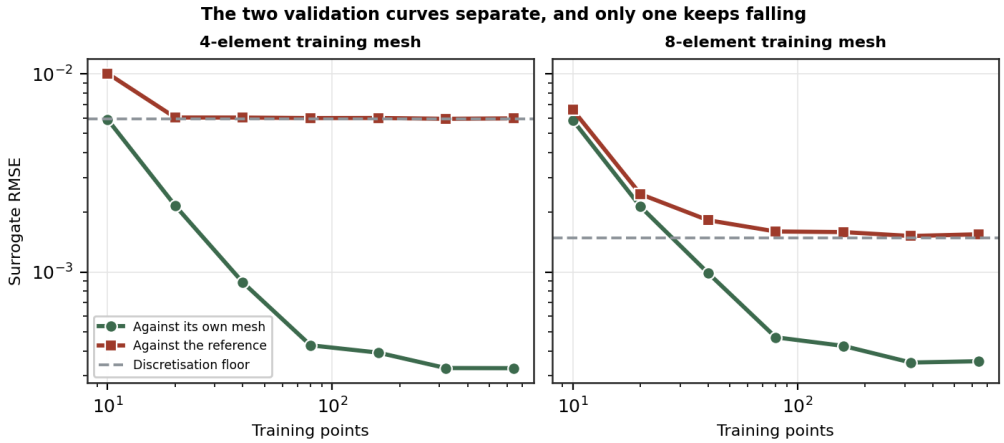


Fig.3: Learning curves scored against the training mesh and against the reference.

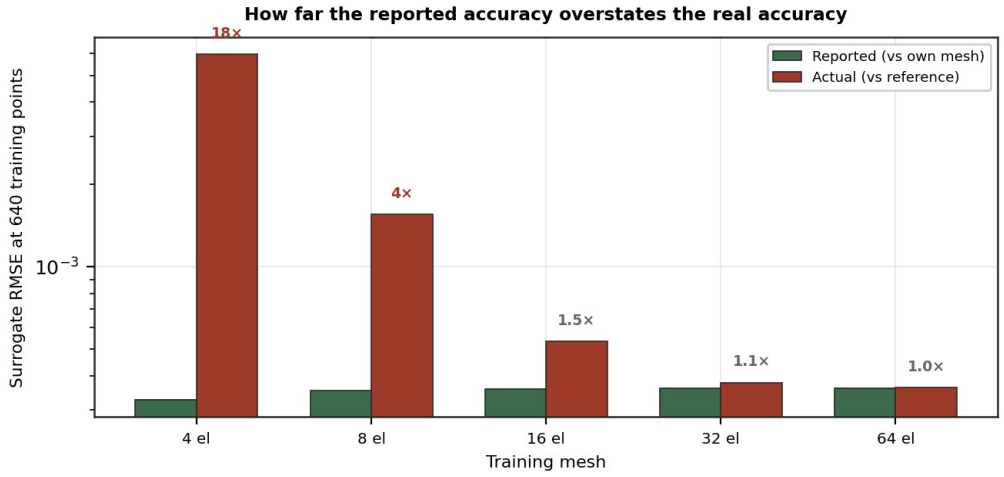


Fig.4: Reported and actual surrogate error at the largest training-set size.

Table 3: Reported against actual error at 640 training points.

Training mesh	RMSE vs own mesh (reported)	RMSE vs reference (actual)	Ratio	Discretisation floor
4 elements	3.27e-4	5.96e-3	18.2	5.93e-3
8 elements	3.55e-4	1.55e-3	4.4	1.48e-3
16 elements	3.59e-4	5.35e-4	1.5	3.71e-4
32 elements	3.60e-4	3.78e-4	1.1	9.29e-5
64 elements	3.60e-4	3.63e-4	1.0	2.32e-5

3.3 More Data Does Not Help

Table 4 records where the learning curve against the reference flattens. On the 4-element mesh the surrogate reached within 10 per cent of its floor after 20 training points; on the 8-element mesh after 80. Beyond those points, the remaining 620 and 560 samples respectively bought nothing in agreement with the physics.

This is the ordinary bias-variance decomposition, but it has an unusual consequence here because the bias is invisible to the practitioner. In a standard setting a plateau in the

learning curve is itself the signal to stop collecting data. Here the curve the practitioner observes — against the training mesh — does not plateau, so the signal to stop never arrives, and simulation budget continues to be spent on a quantity that cannot improve.

The economic implication is direct. Given a fixed simulation budget on a coarse mesh, a developer past the saturation point would be strictly better off spending the remainder on mesh refinement for a subset of points than on additional coarse samples.

Table 4: Saturation of the learning curve against the reference.

Training mesh	Discretisation floor	Saturates at	Final error vs reference	Final error vs own mesh
4 elements	5.93e-3	20 points	5.96e-3	3.27e-4
8 elements	1.48e-3	80 points	1.55e-3	3.55e-4
16 elements	3.71e-4	not reached	5.35e-4	3.59e-4
32 elements	9.29e-5	not reached	3.78e-4	3.60e-4
64 elements	2.32e-5	not reached	3.63e-4	3.60e-4

Saturation is the smallest training size at which error against the reference falls within 10% of the discretisation floor.

3.4 Two Floors, and Which One Binds

Table 5 and Figure 5 separate the two limits. The discretisation floor falls with mesh refinement; the surrogate's own capacity floor — the error a degree-4 polynomial can achieve against any target in this design space — is approximately constant at 3.60e-4.

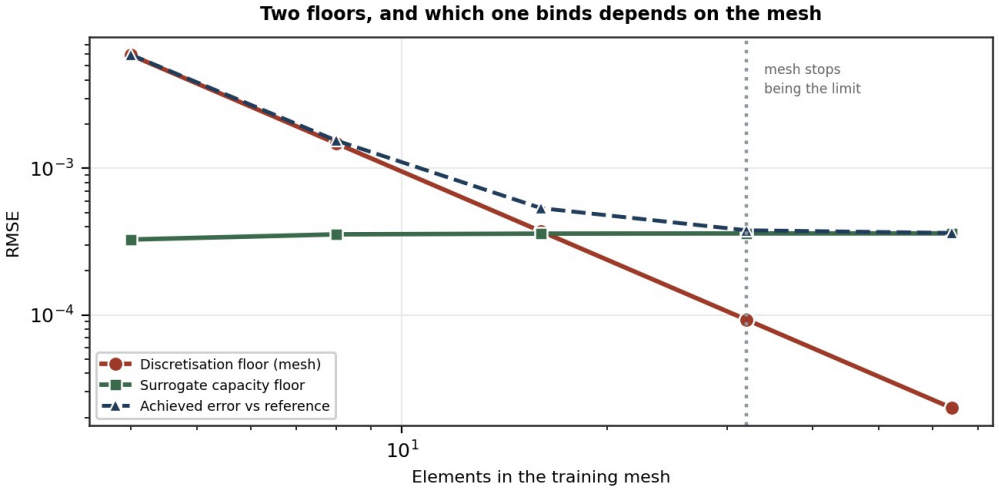


Fig.5: The discretisation floor, the surrogate capacity floor, and the error achieved.

The two cross between 8 and 16 elements. Below the crossing the mesh dominates and refinement is the productive move; above it the surrogate dominates and further refinement changes nothing, as Table 3 shows for the 32- and 64-element rows where the actual error is essentially the capacity floor.

We report this because it qualifies our own argument. The headline concern — that surrogates inherit discretisation error — applies specifically to the regime where the mesh is the larger of the two error terms. A practitioner who has refined past the crossing has a different problem, and would gain from a more expressive surrogate rather than a finer mesh.

Distinguishing the two requires exactly the converged reference that conventional validation omits.

Table 5: Which floor binds, by mesh.

Training mesh	Discretisation floor	Capacity floor	Binding term	Achieved
4 el	5.93e-3	3.27e-4	Mesh	5.96e-3
8 el	1.48e-3	3.55e-4	Mesh	1.55e-3
16 el	3.71e-4	3.59e-4	Mesh	5.35e-4
32 el	9.29e-5	3.60e-4	Surrogate	3.78e-4
64 el	2.32e-5	3.60e-4	Surrogate	3.63e-4

3.5 A Small Converged Budget Removes Most of the Bias

Table 6 and Figure 6 test the remedy. A surrogate fitted to 640 coarse 8-element samples had an error of 1.53e-3 against the reference. Adding 5 converged samples and fitting a discrepancy correction reduced this to 5.11e-4, a factor of 3.0, at 1.13 times the solver cost.

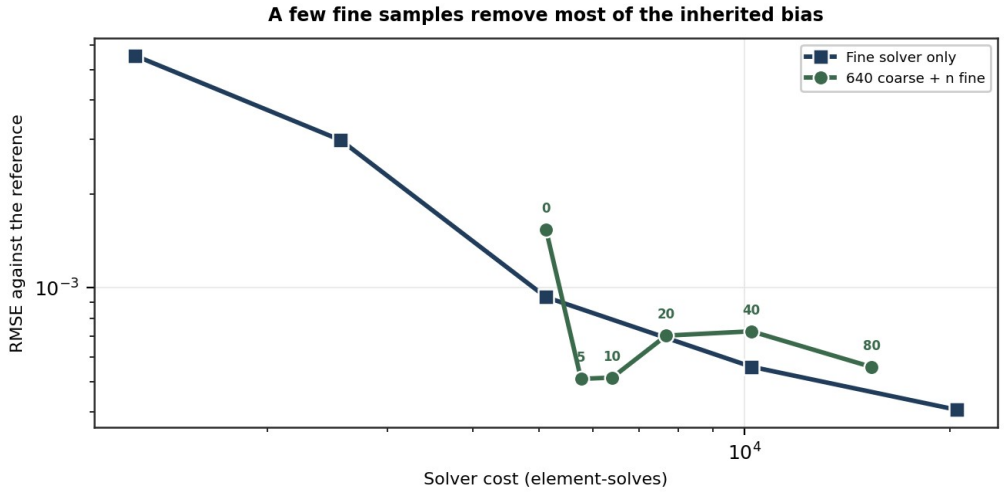


Fig.6: Multi-fidelity correction against a matched-cost single-fidelity baseline.

Against a single-fidelity baseline trained only on converged solves, the comparison is favourable but not dramatic. At a cost of about 5,000 element-solves the multi-fidelity scheme achieved 5.11e-4 while the fine-only surrogate achieved 9.34e-4. The fine-only surrogate eventually overtakes it, reaching 4.07e-4 at 20,480 element-solves, roughly 3.6 times the multi-fidelity cost.

We note an irregularity rather than smoothing it over. The multi-fidelity error is not monotone in the number of converged samples: it is lowest at 5 and rises again at 20 and 40 before falling. With so few points the discrepancy fit is itself high-variance, and the ordering among these configurations should not be over-read. The robust claim is that a handful of converged samples removes most of the inherited bias; the precise optimum is not resolved by this experiment.

Table 6: Multi-fidelity correction and single-fidelity baseline.

Configuration	Converged samples	RMSE vs reference	Solver cost (element-solves)
640 coarse + 0 fine	0	1.53e-3	5,120
640 coarse + 5 fine	5	5.11e-4	5,760
640 coarse + 10 fine	10	5.16e-4	6,400
640 coarse + 20 fine	20	7.03e-4	7,680
640 coarse + 40 fine	40	7.26e-4	10,240
640 coarse + 80 fine	80	5.58e-4	15,360
10 fine only	10	5.54e-3	1,280
20 fine only	20	2.97e-3	2,560
40 fine only	40	9.34e-4	5,120
80 fine only	80	5.59e-4	10,240
160 fine only	160	4.07e-4	20,480

4. DISCUSSION

4.1 Principal Findings

Four findings follow. A surrogate trained on a 4-element mesh reported an error 18 times smaller than its actual disagreement with the converged solution. Error against the physics stopped improving after 20 training points, while the error the practitioner observes continued to fall, removing the usual signal to stop collecting data. The binding error term switched from the mesh to the surrogate's own capacity between 8 and 16 elements. And a discrepancy correction using 5 converged samples reduced the inherited bias by a factor of 3.0.

The underlying point is about what a validation set can establish. Held-out data certify that a model generalises to data drawn from the same process that produced the training data. Where that process contains a systematic error, the certificate is unaffected by it, and the practice of reporting held-out error as though it were accuracy against reality is unsound in exactly the way this experiment isolates.

4.2 Relation to Existing Work

Verification and validation practice in computational mechanics addresses discretisation error directly, through grid convergence studies and formal error estimators, and the distinction between verification and validation is long established [4-7,16,17]. Multi-fidelity surrogate modelling, co-kriging and discrepancy correction are mature techniques developed precisely to combine cheap and expensive model evaluations [8-11], and uncertainty quantification frameworks commonly include numerical error as a contributor [12,13].

Our contribution is not a new method. It is a quantification of how far conventional machine-learning validation can overstate surrogate accuracy when the two traditions are combined carelessly, and an identification of the regime in which the concern applies. The mechanism is known to specialists in both fields; what we have not seen stated with numbers is how large the gap becomes and that the learning curve gives no warning of it.

4.3 Recommendations

Figure 7 and Table 7 set out four recommendations. Two deserve comment.

Report surrogate accuracy against a convergence study, not against held-out solver output. This requires solving a modest number of test points on a refined mesh, and it converts

an uninformative number into an informative one. Where the budget will not stretch to that, the honest report states the training mesh and the convergence status of the underlying solver alongside the held-out error, so that a reader can judge what the number covers.

Spend the marginal simulation on refinement, not on more coarse samples, once the learning curve against a reference has flattened. Identifying that point requires the reference, which is the same investment; but the return is that the remaining budget goes to the term that is actually limiting accuracy.

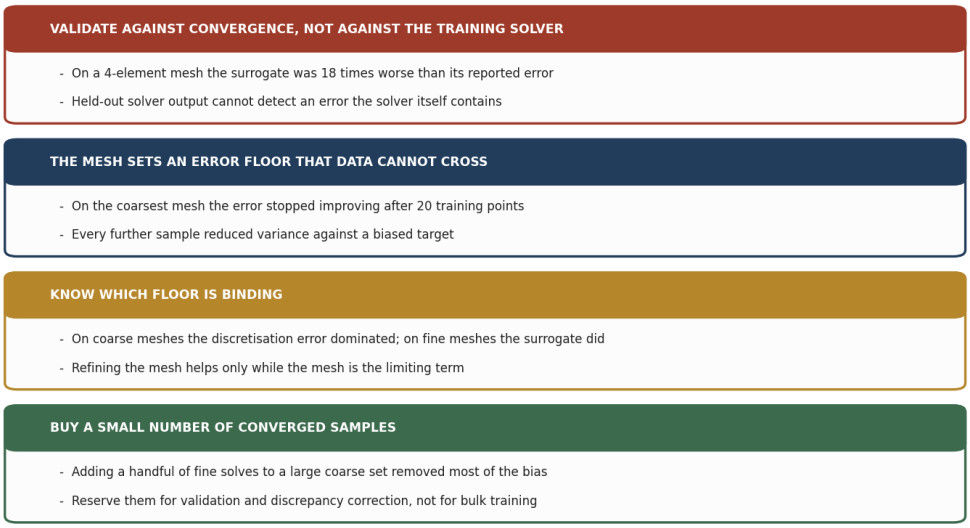


Fig.7: Proposed practice for building and reporting simulation surrogates.

Table 7: Recommendations.

Recommendation	Addresses	Cost	Evidence here
Validate against refined-mesh solutions	Held-out error blind to solver bias	A small refined test set	Section 3.2
Report the training mesh and its convergence status	Accuracy figures without provenance	None	Section 3.2
Stop adding coarse samples once the reference curve flattens	Budget spent on irreducible bias	Requires a reference set	Section 3.3
Establish which floor binds before refining	Refinement that cannot help	One convergence study	Section 3.4
Use a few converged samples for discrepancy correction	Inherited bias at modest cost	Small fine-solve budget	Section 3.5
Prefer fixed-capacity surrogates when diagnosing	Capacity and discretisation error confounded	Interpretability over flexibility	Section 2.2

5. LIMITATIONS

The governing problem is one-dimensional, linear and elliptic, with a smooth three-parameter design space and a scalar quantity of interest. Real engineering surrogates address nonlinear, multi-physics problems in higher dimensions with field outputs, where discretisation error is neither uniform across the design space nor smoothly parameterised. The qualitative mechanism transfers; the specific factors do not.

We used a polynomial surrogate of fixed degree so that capacity could be separated from discretisation. A neural network would have variable and unstated capacity, which would confound the two floors identified in Section 3.4 and would likely make the diagnosis harder rather than changing the underlying effect.

The reference is a 4096-element numerical solution, not an analytic one, so what we call the converged answer carries its own residual error. That error is far below the differences we report and does not affect the conclusions, but the study verifies convergence rather than correctness.

The multi-fidelity result is modest and, as noted in Section 3.5, non-monotone in the number of converged samples. We have not tuned the discrepancy model, compared against co-kriging or other established multi-fidelity methods, or characterised the variance of the estimate across replicates, so it should be read as a demonstration that the approach works rather than as a comparison of methods.

Finally, the discretisation error here is systematic and smooth in the design parameters, which is why a polynomial surrogate reproduces it so faithfully. Where the solver error varies erratically — with mesh-feature interactions or near geometric singularities — the surrogate would fit it less well, and the failure mode would shift from clean bias toward noise that the surrogate cannot represent.

6. CONCLUSION

A polynomial surrogate trained on 4-element finite element data reproduced that data to 3.27e-4 and disagreed with the converged solution by 5.96e-3, a factor of 18. Conventional validation reports the first number.

Error against the physics stopped improving after 20 training points, while the observable learning curve kept falling, so the usual signal to stop spending simulation budget never appeared. Between 8 and 16 elements the binding error term switched from the mesh to the surrogate itself, which changes what the productive next investment is.

Adding 5 converged solves to 640 coarse ones recovered most of the lost accuracy for 13 per cent more solver cost. The general principle is that a held-out set inherits every systematic error of the process that generated it, so a surrogate's certificate of accuracy is only ever as good as the simulations behind it — and reporting which mesh those ran on costs nothing.

DECLARATIONS

Ethics approval and consent to participate. Not applicable. This is a computational study using synthetic data generated by the author. It did not involve human participants, animals or personal data, and institutional review board approval and informed consent were therefore not required.

Consent for publication. Not applicable.

Data availability statement. No external data were used. All results were generated by the solver and surrogate script described in Section 2, developed by the author in Python using NumPy, SciPy and Matplotlib [28-30], with all random seeds fixed. Every parameter is stated in full in Table 1, and every figure and every value in Tables 2 to 6 can be regenerated exactly by executing that script. The script is available from the corresponding author upon reasonable request and will be deposited in a public repository with a permanent identifier on acceptance.

Conflict of interest. The author declares no competing interests, financial or non-financial, in relation to the subject matter of this manuscript. The author has no financial or advisory relationship with any vendor of simulation, surrogate modelling or optimisation software.

Funding. This research received no specific grant from any funding agency in the public, commercial or not-for-profit sectors.

Author's contribution. Proshanta Sarkar is the sole author of this work and was responsible for the conception and design of the study, the implementation of the solver and surrogate code, the design and execution of the experiments, the analysis and verification of all reported values, the preparation of the figures and tables, the formulation of the recommendations, and the drafting, revision and final approval of the manuscript. The author is accountable for all aspects of the work.

Acknowledgements. None declared.

REFERENCES

1. Forrester AIJ, Sobester A, Keane AJ. Engineering design via surrogate modelling: a practical guide. Chichester: Wiley; 2008.
2. Queipo NV, Haftka RT, Shyy W, Goel T, Vaidyanathan R, Tucker PK. Surrogate-based analysis and optimization. Prog Aerosp Sci. 2005;41(1):1-28.
3. Kudela J, Matousek R. Recent advances and applications of surrogate models for finite element method computations: a review. Soft Comput. 2022;26(24):13709-33.
4. Roache PJ. Verification and validation in computational science and engineering. Albuquerque: Hermosa; 1998.
5. Oberkampf WL, Roy CJ. Verification and validation in scientific computing. Cambridge: Cambridge University Press; 2010.
6. Roache PJ. Perspective: a method for uniform reporting of grid refinement studies. J Fluids Eng. 1994;116(3):405-13.
7. American Society of Mechanical Engineers. Standard for verification and validation in computational fluid dynamics and heat transfer. ASME V&V 20-2009. New York: ASME; 2009.
8. Kennedy MC, O'Hagan A. Predicting the output from a complex computer code when fast approximations are available. Biometrika. 2000;87(1):1-13.
9. Kennedy MC, O'Hagan A. Bayesian calibration of computer models. J R Stat Soc Series B. 2001;63(3):425-64.
10. Peherstorfer B, Willcox K, Gunzburger M. Survey of multifidelity methods in uncertainty propagation, inference, and optimization. SIAM Rev. 2018;60(3):550-91.
11. Fernandez-Godino MG, Park C, Kim NH, Haftka RT. Review of multi-fidelity models. arXiv:1609.07196. 2016.
12. Smith RC. Uncertainty quantification: theory, implementation, and applications. Philadelphia: SIAM; 2013.
13. Sullivan TJ. Introduction to uncertainty quantification. Cham: Springer; 2015.
14. Brenner SC, Scott LR. The mathematical theory of finite element methods. 3rd ed. New York: Springer; 2008.
15. Hughes TJR. The finite element method: linear static and dynamic finite element analysis. Mineola: Dover; 2000.
16. Ainsworth M, Oden JT. A posteriori error estimation in finite element analysis. New York: Wiley; 2000.
17. Babuska I, Rheinboldt WC. A posteriori error estimates for the finite element method. Int J Numer Methods Eng. 1978;12(10):1597-615.
18. Bathe KJ. Finite element procedures. 2nd ed. Watertown: Bathe; 2014.
19. Raissi M, Perdikaris P, Karniadakis GE. Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J Comput Phys. 2019;378:686-707.

20. Lu L, Jin P, Pang G, Zhang Z, Karniadakis GE. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. Nat Mach Intell. 2021;3(3):218-29.
21. Li Z, Kovachki N, Azizzadenesheli K, Liu B, Bhattacharya K, Stuart A, et al. Fourier neural operator for parametric partial differential equations. Int Conf Learn Represent. 2021.
22. Kim Y, Choi Y, Widemann D, Zohdi T. A fast and accurate physics-informed neural network reduced order model with shallow masked autoencoder. J Comput Phys. 2022;451:110841.
23. Willard J, Jia X, Xu S, Steinbach M, Kumar V. Integrating scientific knowledge with machine learning for engineering and environmental systems. ACM Comput Surv. 2022;55(4):1-37.
24. Karniadakis GE, Kevrekidis IG, Lu L, Perdikaris P, Wang S, Yang L. Physics-informed machine learning. Nat Rev Phys. 2021;3(6):422-40.
25. Sacks J, Welch WJ, Mitchell TJ, Wynn HP. Design and analysis of computer experiments. Stat Sci. 1989;4(4):409-23.
26. Santner TJ, Williams BJ, Notz WI. The design and analysis of computer experiments. 2nd ed. New York: Springer; 2018.
27. Gramacy RB. Surrogates: Gaussian process modeling, design, and optimization for the applied sciences. Boca Raton: Chapman and Hall/CRC; 2020.
28. Harris CR, Millman KJ, van der Walt SJ, Gommers R, Virtanen P, Cournapeau D, et al. Array programming with NumPy. Nature. 2020;585(7825):357-62.
29. Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, et al. SciPy 1.0: fundamental algorithms for scientific computing in Python. Nat Methods. 2020;17(3):261-72.
30. Hunter JD. Matplotlib: a 2D graphics environment. Comput Sci Eng. 2007;9(3):90-95.