FROM PHYSICS-INFORMED LOSSES TO EQUATION-DERIVED PRIORS AND REPRESENTATIONS FOR NEURAL OPERATORS

Anonymous authors
Paper under double-blind review

ABSTRACT

When governing equations are known but solution data are scarce, equation knowledge can be used not only as a training constraint but also as an inductive bias for the neural operator. We study three ways of exposing equation information: enforcing the equation as a loss, using it to construct an equation-derived prior, and injecting the structure it reveals into the residual. Our idea is to use the governing equation to reduce the learning problem before training. Across the benchmarks that admit a physics-informed loss, the equation-derived prior compares favorably with imposing the same equation as a training loss, and some problems benefit further from structural injection beyond the prior. These results suggest that equation knowledge is more effective when it is used to shape the representation before training than when used solely as a training constraint.

1 INTRODUCTION

Numerical solvers for partial differential equations (PDEs) are central to computational science and engineering, but each run solves a single instance. Design optimization, uncertainty quantification, inverse problems and control all require the same equation to be solved repeatedly across a family of parameters, geometries or boundary conditions, and the cost of rerunning a high-fidelity solver for every query quickly dominates (Forrester & Keane, 2009; Psaros et al., 2023).

Neural operators are the standard surrogate for this setting: rather than approximating a single solution, they learn a map from parametric inputs to solution fields that generalizes across the family (Lu et al., 2021; Lee et al., 2023; Li et al., 2021; Hao et al., 2023; Wang et al., 2025). Their accuracy depends on the amount of labeled solution data, and every label is itself a high-fidelity solve, so labeled data are most expensive in precisely the settings where a surrogate is most needed.

When a governing equation is known, even approximately, it can reduce the amount of solution data required for learning. The standard route is to add its residual to the training objective (Raissi et al., 2019): the network is penalized for violating the equation while still representing the entire solution field, and the same idea carries over to operator learning (Li et al., 2024; Wang et al., 2021). This reduces the labeled-data requirement, but the equation enters only through the objective, and even where it is exact, a lower residual loss does not guarantee a more accurate solution (Chen et al., 2026).

An alternative is to give the learner more than a loss, and prior work does so in three ways. A data-derived reference gives the learner a starting point to correct: a reduced basis from snapshots (Sirovich, 1987; Barrault et al., 2004; Chaturantabut & Sorensen, 2010), which POD-DeepONet places in the trunk of the operator (Lu et al., 2022), a low-fidelity solution (Meng & Karniadakis, 2020), or a retrieved neighbor whose residual is learned (Yue et al., 2025). Model-form correction starts instead from an approximate simulator and learns its discrepancy, classically with a Gaussian process (Kennedy & O'Hagan, 2001; Brynjarsdóttir & O'Hagan, 2014) and recently with a second operator trained under the misspecified residual (Ma et al., 2026a). A third line keeps the approximate equation inside training, as a solver in the loop (Um et al., 2020; Kochkov et al., 2021), as simplified forms trained jointly (Ma et al., 2026b), or as an architecture aligned to structure found in data (Roohi & Mahdavi, 2026; Dang et al., 2026). These approaches typically rely on learned, data-derived, or simulator-provided references, rather than asking what reusable representation can be extracted directly from an approximate equation before training: a solvable field, an invariant,

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

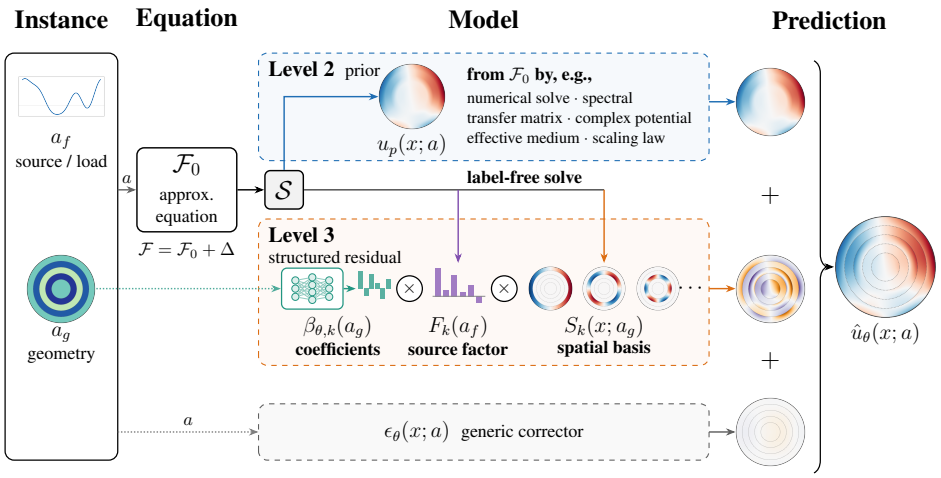

Figure 1: **Overview of the inference pipeline.** Given an instance $a = (a_f, a_g)$, the construction $\mathcal{S}$ uses the approximate equation $\mathcal{F}_0$ to derive the prior u_p and the residual basis $\Phi_{\mathcal{F}_0,k} = F_k(a_f)\, S_k(x; a_g)$, without requiring solution labels. A learned branch $\beta_\theta(a_g)$ predicts the basis coefficients, while a generic corrector ϵ_θ captures the remaining discrepancy. The prior, structured residual, and generic correction are then combined to give $\hat{u}_\theta$ (Eq. 14). Levels 2–3 of the framework are shown; Level 1 uses the equation only as a training constraint.

a low-dimensional instance factor, a basis, or a coordinate transformation, all obtained without labels or a learned reference. Section 2 unifies these uses in a single factorization, while Appendix K discusses the closest related approaches.

The equation can enter the learning problem at three levels, depending on what the construction exposes. Writing the progression as

$$\mathcal{F}_0 \;\longrightarrow\; u_p + r \;\longrightarrow\; u_p + \Phi_{\mathcal{F}_0}\beta_\theta + \epsilon_\theta, \tag{1}$$

the equation enters as a *constraint* through a residual penalty, as a *prior* u_p that fixes part of the solution map before training, and as a *representation* $\Phi_{\mathcal{F}_0}$ in which the remaining residual is learned; we refer to these as Levels 1, 2 and 3. The question we study is how much of a neural operator's target an approximate equation can resolve before training, and when the remainder admits a reusable representation. Here, $r = u - u_p$ denotes the residual left after removing the equation-derived prior (Eq. 7); when the construction exposes reusable structure, the part of r in the span of $\Phi_{\mathcal{F}_0}$ is recovered through learned coefficients β_θ, and a generic learned corrector ϵ_θ absorbs the remainder (Eq. 12). Our main contributions are as follows:

- We provide a unified view of equation information in neural operator learning through three roles (constraint, prior, and residual representation) and empirically show that where the same equation information enters the learning problem can substantially affect data efficiency.
- We construct equation-derived priors for eighteen benchmarks, and residual bases for the six benchmarks whose approximate equations expose reusable residual structure, from numerical solves of the approximate equation to spectral, transfer-matrix and complex-potential forms.
- We compare the prior against the same equation used as a loss, at its oracle-selected weight, and against a raw operator on the benchmarks where no matched residual loss can be formed.
- We introduce structural injection, which learns the residual in the equation-derived basis, against two controls of the same width: a random basis and one built from the training residuals by POD.

Figure 1 provides an overview of the inference pipeline.

2 Equation Information as a Representation

We consider a family of problems parameterized by an input $a \in \mathcal{A}$ collecting coefficients, sources, geometry or material data. For each instance the solution $u(x; a)$, at a spatial or query coordinate x of a domain Ω, satisfies a governing equation $\mathcal{F}$ and a boundary condition $\mathcal{B}$,

$$\mathcal{F}(u; a) = 0 \;\text{ in } \Omega, \qquad \mathcal{B}(u; a) = 0 \;\text{ on } \partial\Omega, \tag{2}$$

and we learn an approximation $\mathcal{G}_\theta : \mathcal{A} \to \mathcal{U}$ of the solution operator $\mathcal{G} : a \mapsto u(\cdot\,; a)$, writing $\hat{u}_\theta(x; a) = \mathcal{G}_\theta(a)(x)$, or $\hat{u}$ when θ is clear. Two properties define the setting. Labels are scarce: the sample $\mathcal{D}_N = \{(a_i, u_i)\}_{i=1}^N$ has small N. And $\mathcal{F}$ is known only approximately, through an operator $\mathcal{F}_0$ that is typically the same equation with a simplified coefficient, closure, interface condition or geometry,

$$\mathcal{F}(u; a) = \mathcal{F}_0(u; a) + \Delta(u; a), \tag{3}$$

where Δ is the omitted part, unknown and not assumed small. On most benchmarks it acts in the interior and $\mathcal{B}$ is shared. A simplified interface condition or geometry instead moves it onto interfaces, as on 3D layered and annulus, or onto the boundary itself, as on elasticity, whose prior replaces a finite plate by an infinite one.

Level 1: the equation as a constraint. The standard use adds the residual of $\mathcal{F}_0$, with a weight $\lambda > 0$, to the supervised objective (Raissi et al., 2019; Li et al., 2024; Wang et al., 2021),

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} \big\|\hat{u}_\theta(\cdot\,; a_i) - u_i\big\|^2 + \frac{\lambda}{N}\sum_{i=1}^{N} \big\|\mathcal{F}_0(\hat{u}_\theta; a_i)\big\|^2_{L^2(\Omega)}, \tag{4}$$

with the penalty evaluated at collocation points. The target is still the full field u, and where $\mathcal{F}_0$ is wrong the penalty moves it.

Proposition 1 (Population bias of the residual penalty)**.** *Let $\mathcal{F}_0(v) = L_0 v - f$ with L_0 linear. Over all fields v, the minimizer of $\|v - u\|^2 + \lambda\|\mathcal{F}_0(v)\|^2$ is $v_\lambda = u + \lambda\big(I + \lambda L_0^\top L_0\big)^{-1} L_0^\top \Delta(u)$.*

The displacement vanishes only when $\Delta(u) = 0$ and does not involve N: more labels do not remove it, and λ sets its size. This concerns the population minimizer; a finite-sample estimator need not reach v_λ (Appendix A).

Levels 2 and 3: the equation before training. The alternative uses $\mathcal{F}_0$ before training, through a construction $\mathcal{S}$ that is cheap relative to training and consumes no solution labels, with one exception (Section 4.3). It changes what is learned rather than how: what $\mathcal{F}_0$ determines is removed from the target, labels are spent only on the remainder, and Δ is left for data to correct rather than imposed on the fit. In our benchmarks the construction returns one of three objects. A *direct prior* is a field solved from $\mathcal{F}_0$, by a numerical solve, a spectral expansion, a transfer matrix, an effective-medium coefficient or a complex potential,

$$u_p(\cdot\,; a) = \mathcal{S}(\mathcal{F}_0, a). \tag{5}$$

An *indirect prior* starts from an instance-side factor $\psi_{\mathcal{F}_0}(a) \in \mathbb{R}^d$, a few numbers per instance that the equation fixes, such as $q(a)\,(1, \alpha)^\top$ for an airfoil with dynamic pressure q and incidence α (Section 4.3). Only its spatial profile $c(x) \in \mathbb{R}^d$ is fitted, by least squares over the N labeled instances at each query point,

$$u_p(x; a) = c(x)^\top \psi_{\mathcal{F}_0}(a), \qquad c(x) = \arg\min_{c \in \mathbb{R}^d} \sum_{i=1}^{N} \big(c^\top \psi_{\mathcal{F}_0}(a_i) - u_i(x)\big)^2, \tag{6}$$

so the labels determine d numbers per query point and $N \geq d$ instances suffice. Either prior leaves the residual

$$r(x; a) = u(x; a) - u_p(x; a), \tag{7}$$

for learning. For a direct prior this residual is not an arbitrary leftover.

Lemma 1 (Residual identity)**.** *Let $\mathcal{F}_0(v) = L_0 v - f$ with L_0 linear and invertible under the boundary condition u and u_p share, and let $\mathcal{F}_0(u_p) = 0$. Then $r = -L_0^{-1}\Delta(u)$.*

The residual is the known inverse operator applied to the omitted term, so its form follows from where Δ acts. An interior term expands in the eigenbasis (ϕ_m, κ_m) of L_0 with coefficients damped by κ_m^{-1}: on 2D Poisson the prior drops μu from $-\nabla^2 u + \mu u = f$, and with $\hat{f}_m$ the coefficients of f, on the modes it keeps $r = \sum_m \hat{f}_m\big[(\kappa_m + \mu)^{-1} - \kappa_m^{-1}\big]\phi_m$. A term on interfaces or on the outer boundary, as on 3D layered, annulus and elasticity, lies outside the lemma, but r then solves the homogeneous equation $L_0 r = 0$ away from them. A *representation* supplies that family as a basis $\Phi_{\mathcal{F}_0}(x; a)$ for the residual, with coefficients $\beta(a)$ and a remainder ϵ outside its span,

$$r(x; a) = \Phi_{\mathcal{F}_0}(x; a)\,\beta(a) + \epsilon(x; a), \tag{8}$$

Table 1: **The four slots of Eq. 10.** Each use of $\mathcal{F}_0$ studied here fills one slot and leaves the rest at their data-driven defaults; with every slot at its default the model is a raw operator. The objective row is Proposition 1 written as a target and a metric.

Slot	Default	Filled by the equation	Use	Level
objective $(u^\dagger, M)$	$(u,\ I)$	$(v_\lambda,\ I + \lambda L_0^\top L_0)$	residual loss	1
anchor u_0	0	u_p	direct prior	2
instance side ψ	learned	$\psi_{\mathcal{F}_0}$	indirect prior	2
query side φ	learned	$\Phi_{\mathcal{F}_0}$	representation	3

such as the modal basis of a spectral expansion, the layer-wise solutions of a transfer matrix, the Laurent basis of a conformal map, or, where the omitted term is not differential, the sensitivity directions of the prior. The three outputs are therefore

$$\begin{aligned} &\text{direct prior:}\quad \mathcal{S}(\mathcal{F}_0, a) \longrightarrow u_p, \qquad \text{indirect prior:}\quad \psi_{\mathcal{F}_0} \xrightarrow{\text{fit from data}} u_p, \\ &\text{representation:}\quad \Phi_{\mathcal{F}_0} \xrightarrow{\text{learn coefficients}} \beta_\theta, \end{aligned} \tag{9}$$

the two priors realizing Level 2 of Eq. 1 and the representation Level 3. Table 5 in Appendix B names the construction and output for every benchmark, and the subsections there derive them.

One factorization. The three levels can be compared inside the branch–trunk form the backbone already has. Write the prediction as an anchor field u_0 plus the inner product of an instance-side factor $\psi(a)$, produced by the branch, and a query-side factor $\varphi(x; a)$, produced by the trunk, with every learned part trained jointly to fit a target $u^\dagger$ in a metric M:

$$\hat{u}(x; a) = u_0(x; a) + \psi(a)^\top \varphi(x; a), \qquad \min_\theta\ \mathbb{E}_a \big\| \hat{u}(\cdot\,; a) - u^\dagger(\cdot\,; a) \big\|_M^2. \tag{10}$$

Each use we study fills one of these four slots (Table 1), and leaving all four at their data-driven defaults gives a raw operator.

The objective row restates Proposition 1: completing the square in Eq. 4 turns the penalty into a fit to v_λ in the metric M, which reweights the error and, unless $\Delta(u) = 0$, also moves the target. The indirect prior alone consumes labels because the equation fixes one factor and the data fit the other. A model may combine slots, and may fill a slot only up to a learned factor, the equation supplying a multiplier and the network the rest (Sections 3 and 4.3). Appendix H compares the equation anchor with five data-derived ones. The factorization is used to compare formulations, not to claim that every use of an equation occupies exactly one slot.

3 Equation-Derived Neural Operator Formulations

Prior with a generic corrector. The Level 2 model adds a learned corrector to either prior,

$$\hat{u}_\theta(x; a) = u_p(x; a) + r_\theta(x; a), \qquad r_\theta(x; a) = \sum_{i=1}^{p} b_{\theta,i}(a)\, t_{\theta,i}(x), \tag{11}$$

where r_θ is the backbone operator, written here in the DeepONet form with branch b_θ and trunk t_θ; with an FNO backbone it is the FNO itself. Under DeepONet the last linear layer of the corrector is initialized to zero, so training starts at u_p up to the mean of the training residuals, and labels are spent only on the residual r of Eq. 7. Under FNO the corrector keeps its standard initialization: zero-initializing its last layer starts the body with no data gradient, and weight decay drives it to zero (Appendix C.2).

Structural injection. In DeepONet terms, structural injection replaces part of the learned trunk representation with basis functions derived directly from the approximate equation, leaving the network to learn primarily their instance-dependent coefficients and the remaining correction.

When the construction exposes a representation, the Level 3 model learns r in the form of Eq. 8, so that the part in the span of $\Phi_{\mathcal{F}_0}$ is recovered by estimating coefficients rather than by learning a spatial form,

$$\hat{u}_\theta(x; a) = u_p(x; a) + \sum_{k=1}^{n_b} \beta_{\theta,k}\, \Phi_{\mathcal{F}_0,k}(x; a) + \epsilon_\theta(x; a), \tag{12}$$

where β_θ is a learned map onto the coefficients β, and ϵ_θ, the learned counterpart of ϵ, is the corrector r_θ of Eq. 11 in its DeepONet form, retained unchanged; nothing confines it to the complement of

Table 2: **Prior against residual loss at N=200.** Relative L_2 error, mean $\pm$ sample standard deviation over five seeds; bold marks the row minimum. $\varepsilon_{\text{prior}}$ is the error of u_p alone and PI is at its oracle weight; "–" marks the rows admitting no matched residual loss. All budgets: Table 11.

Benchmark	Backbone	$\varepsilon_{\text{prior}}$	RAW	PI	PRIOR
a differentiable residual of $\mathcal{F}_0$ is available					
1D nonlinear reaction	DeepONet	0.0029	0.0246 $\pm$0.0029	0.0230 $\pm$0.0024	**0.0004** $\pm$0.0001
1D nonlinear reaction	FNO	0.0029	0.0064 $\pm$0.0007	0.0051 $\pm$0.0004	**0.0006** $\pm$0.0001
Darcy (Li et al., 2021)	DeepONet	0.2511	0.0989 $\pm$0.0040	0.0950 $\pm$0.0020	**0.0716** $\pm$0.0007
Darcy (Li et al., 2021)	FNO	0.2511	0.0185 $\pm$0.0010	**0.0184** $\pm$0.0010	0.0193 $\pm$0.0011
no matched residual loss can be formed					
Pipe (Li et al., 2023)	DeepONet	0.1070	0.1009 $\pm$0.0058	–	**0.0434** $\pm$0.0025
Pipe (Li et al., 2023)	FNO	0.1070	**0.0169** $\pm$0.0015	–	0.0196 $\pm$0.0019
Pipe-GRF	DeepONet	0.0842	0.1606 $\pm$0.0079	–	**0.0791** $\pm$0.0024
Pipe-GRF	FNO	0.0842	0.0484 $\pm$0.0029	–	**0.0385** $\pm$0.0015

the span. Both start at zero, and $\beta_\theta = 0$ with $\epsilon_\theta = r_\theta$ recovers Eq. 11 exactly. The two branches are trained in stages within one loss,

$$\mathcal{L}_{\text{struct}} = \ell\big(u_p + \Phi_{\mathcal{F}_0}\beta_\theta\big) + \ell\big(u_p + \text{sg}[\Phi_{\mathcal{F}_0}\beta_\theta] + \epsilon_\theta\big), \tag{13}$$

where ℓ is the relative L_2 error against u and sg stops the gradient: the structured branch fits the whole residual and ϵ_θ only what it leaves, the division of Eq. 8. A joint loss on the sum lets the two branches grow and cancel. The prediction is still Eq. 12.

Instance and spatial factors. We write $a = (a_f, a_g)$, where a_f collects the source, boundary or load data on which r depends linearly and a_g the parameters that fix the domain or its layout, and factor each column as $\Phi_{\mathcal{F}_0,k}(x; a) = F_k(a_f)\, S_k(x; a_g)$: the source factor F_k is a coefficient of a_f in the basis the construction uses, and the spatial factor S_k is the field it multiplies. Eq. 12 becomes

$$\hat{u}_\theta(x; a) = u_p(x; a) + \sum_{k=1}^{n_b} \beta_{\theta,k}(a_g)\, F_k(a_f)\, S_k(x; a_g) + \epsilon_\theta(x; a). \tag{14}$$

In the branch–trunk form, β_θ is the branch and $\Phi_{\mathcal{F}_0} = FS$ takes the place of the learned trunk; neither F nor S is trained. Because F carries all dependence on a_f, the coefficients β depend on a only through a_g, and so does β_θ. It is a learned constant vector on 2D Poisson and optical, and $F_k \equiv 1$ when r has no linear instance data. Appendix B.8 gives F, a_g and S for each benchmark, in one of five forms: a fixed eigenbasis, a separable basis, geometry-dependent fields shared by every coefficient of a_f, a sensitivity of the prior, and a response of the prior operator itself.

Because the structured branch adds parameters, Section 4.1 compares it with capacity-matched controls that replace S (Appendix C.2); its relation to POD-DeepONet (Lu et al., 2022) is in Appendix K.

4 RESULTS

Of the eighteen benchmarks, eleven admit a differentiable residual in $\mathcal{F}_0$ and carry the loss-versus-prior comparison; six expose a reusable representation $\Phi_{\mathcal{F}_0}$ and carry the structural-injection experiments. Table 7 in Appendix C lists all eighteen with the formulations each admits.

4.1 EXPERIMENT 1: EQUATION-DERIVED PRIORS VS. PHYSICS LOSSES

We ask whether the governing PDE is a stronger inductive bias as an equation-derived prior than as an explicit residual loss. Three formulations carry that comparison: a raw neural operator (RAW), a physics-informed one that adds the residual to the objective (PI), and the equation-derived prior (PRIOR), matched in every respect but where the physics enters. Three further formulations test the representation: structural injection (PRIOR-STRUCT), Eq. 12, and two controls matched to it in width and parameter count: a random basis (PRIOR-RAND) and one built from the training residuals by POD (PRIOR-POD). More details are in Appendix C.

Table 2 shows the prior improving on the raw model far more than the physics-informed loss does, and shows where it does not. On 1D nonlinear reaction, where the construction resolves the field almost completely ($\varepsilon_{\text{prior}} = 0.0029$), the prior is $56\times$ below the oracle-weighted loss under DeepONet and $8.4\times$ below it under FNO; the loss itself moves the raw error by 6% and 20%.

How far it improves depends on how much it resolves, and on how much the backbone already resolves without it. Over the thirteen benchmark–architecture pairs that admit both formulations,

Table 3: **Structural injection against capacity-matched controls at N=200.** Relative L_2 error, DeepONet backbone, mean ± sample standard deviation over five seeds; bold marks the row minimum. PRIOR-RAND and PRIOR-POD replace the spatial factor S (on Pipe, all of $\Phi_{\mathcal{F}_0}$) by a random and a POD basis at the same width and parameter count. All six benchmarks that expose $\Phi_{\mathcal{F}_0}$ are listed; Figure 2 gives the same comparison against the training budget.

Benchmark	u_p alone	PRIOR	PRIOR-STRUCT	PRIOR-RAND	PRIOR-POD
2D Poisson, $N = 200$	0.1827	0.0161 ±0.0006	**0.0025** ±0.0001	0.0179 ±0.0007	0.0134 ±0.0004
3D layered, $N = 200$	0.1751	0.2377 ±0.0096	**0.0347** ±0.0023	0.2633 ±0.0090	0.2253 ±0.0179
Annulus, $N = 200$	0.6649	0.5195 ±0.0204	**0.0251** ±0.0013	0.5568 ±0.0325	0.5738 ±0.0113
Elasticity, $N = 200$	0.3214	0.1098 ±0.0018	**0.0626** ±0.0016	0.1352 ±0.0035	0.2754 ±0.0349
Pipe, $N = 200$	0.1070	0.0434 ±0.0025	**0.0244** ±0.0019	0.0439 ±0.0025	0.0254 ±0.0007
Optical, $N = 200$	0.2941	**0.1229** ±0.0087	0.1258 ±0.0059	0.1369 ±0.0020	0.1252 ±0.0077

the three linear 1D benchmarks set aside, across all budgets, the geometric-mean PI/PRIOR ratio is $2.73\times$ and the prior is the more accurate of the two in 42 of the 63 pair–budget cells. Split at the median of $\varepsilon_{\text{prior}}$, the error of u_p alone, the six pairs below 0.196 give $6.48\times$ with the prior ahead in 30 of 34 cells, while on the seven above the two formulations are indistinguishable in aggregate ($0.99\times$, 12 of 29; Table 13). The excluded three (1D heat, 1D convection, battery) are exactly linear and favor the prior in 29 of 30 cells; including them gives $2.52\times$ and 71 of 93. The oracle picks $\lambda \in [10^{-10}, 1]$ by test error, so these are conservative estimates of the margin under practical λ-selection rules; Appendix C.1 shows that a leave-one-seed-out rule does not reduce it.

At N=200, the four pairs where the loss leads are all Darcy problems under the effective-medium construction (1D Darcy under both backbones, and 2D Darcy and Darcy (Li et al., 2021) under FNO). The absolute margins range from 0.0007 to 0.0062, significant in three cases ($p \leq 0.021$; Appendix C.4) but not for Darcy under FNO ($p = 0.13$). The lower block of Table 2 shows a similar pattern where no matched loss is available: on Pipe, the prior is $2.3\times$ below RAW under DeepONet, which cannot resolve the field from 200 labels alone, but slightly above it under FNO, which can; on Pipe-GRF it helps under both ($2.0\times$ and $1.26\times$). Overall, the equation is more effective as a prior than as a training constraint, but this does not imply that physics always helps (FNO rows of Table 2).

Table 3 shows structural injection improving further where the residual has algebraic structure to expose. The separation that matters is PRIOR-RAND/PRIOR-STRUCT, since a random basis adds capacity alone: $7.18\times$ on 2D Poisson, $7.58\times$ on 3D layered, $22.21\times$ on annulus, $2.16\times$ on elasticity, $1.80\times$ on Pipe and $1.09\times$ on optical. The random basis of matched width is worse than the prior it was added to on five of the six ($p \leq 0.02$) and matches it on Pipe (0.0439 against 0.0434, $p = 0.40$), so the gain is not the extra capacity: it is the basis the equation prescribes. PRIOR-POD, which draws the same width from the training residuals, does not close the gap on the four benchmarks whose residual carries its own structure, and on annulus and elasticity is worse than the random basis (Appendix J). Pipe and optical are the exceptions. On Pipe the shape family is nine-dimensional, so a basis fitted to the training residuals reaches the same place: PRIOR-POD is unresolved against PRIOR-STRUCT at N=200 (0.0254 against 0.0244, $p = 0.21$) and ahead of it at larger budgets (0.0148 against 0.0160 at N=500, $p = 0.02$; 0.0108 against 0.0125 at N=1000, $p = 0.005$); on optical the sensitivities do not span the residual (Appendix F) and PRIOR-STRUCT, PRIOR and PRIOR-POD do not separate ($p \geq 0.6$). 3D layered is the sharpest case (Appendix D): there the generic corrector of Eq. 11 leaves the error *above* u_p alone (0.2377 against 0.1751), while the same corrector with the equation-derived basis reaches 0.0347.

The prior is also the cheaper formulation: constructing it for the test set costs a median of 0.05% of one PRIOR run, whereas the nine-point PI sweep costs 9–19× one PRIOR run (Appendix L).

4.2 EXPERIMENT 2: STRUCTURAL INJECTION UNDER VARYING DATA BUDGETS

A single budget cannot distinguish a model that is genuinely more accurate from one that simply starts from a better initialization but stops improving with more data. We therefore sweep $N \in \{10, 25, 50, 100, 200, 500, 1000\}$, keeping all other settings fixed as in Section 4.1, with five seeds per point.

Figure 2 shows the formulations differing not only in error at a fixed budget but in what additional labels buy. On 3D layered the formulations without the equation-derived basis stall as N grows: RAW gets *worse* as the budget grows (1.14 at N=10 to 1.64 at N=1000), PRIOR drifts up as well (0.191 to 0.264), and PRIOR-STRUCT falls from 0.136 to 0.030. An error above one is worse than

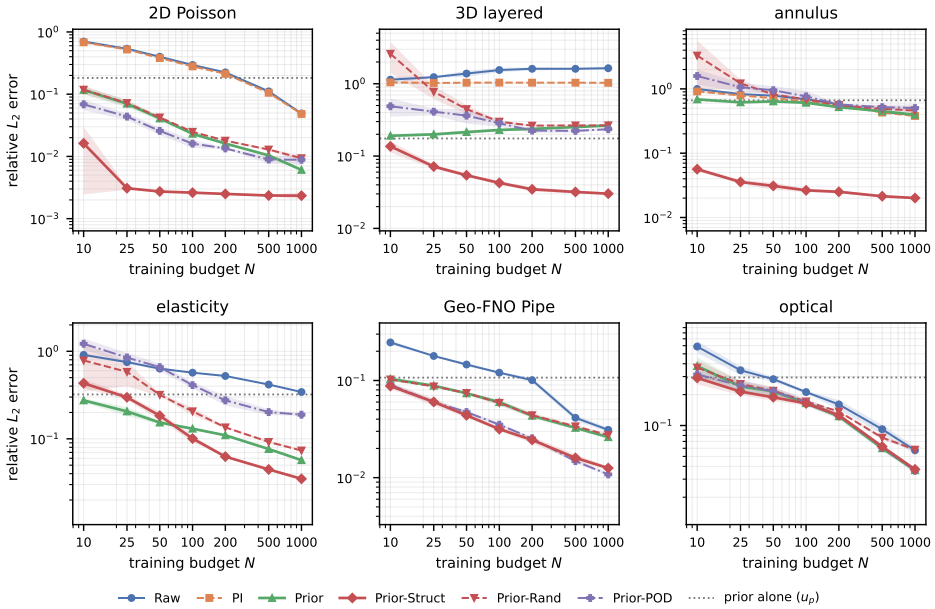

Figure 2: **Error against training budget.** Relative L_2 error against N, both axes logarithmic, DeepONet backbone, for the six benchmarks that expose $\Phi_{\mathcal{F}_0}$. Bands are ± 1 sample standard deviation over five seeds; the dotted line is $\varepsilon_{\text{prior}}$, which does not depend on N. PI is at its oracle weight and only where a residual of $\mathcal{F}_0$ exists, so it is absent on elasticity, Pipe and optical. PRIOR-STRUCT is below the dotted line at every budget except elasticity at $N{=}10$. On 3D layered the other formulations stay above that line as N grows (RAW and PI stay above one), and on annulus they fall below it only slowly; where the residual is low-dimensional PRIOR-POD reaches PRIOR-STRUCT, and on Pipe passes it at $N \geq 500$.

predicting zero: PI collapses toward zero and RAW is full-scale but uncorrelated with the target (Appendix C.3), as 1000 labels still do not reveal the layer interfaces the equation-derived basis supplies. On annulus PRIOR-STRUCT at $N{=}10$ (0.056) is already below every other formulation at $N{=}1000$ (0.379 to 0.503), a hundredfold difference in budget; on the other four, the other formulations improve with data as well.

Two qualifications. Structural injection is not uniformly beneficial at the smallest budgets: on elasticity it does not overtake PRIOR until $N{=}100$, a property of how much of the basis the budget can estimate rather than an optimization accident (Appendix C.5). And the benefit saturates once the residual is captured: on 2D Poisson PRIOR-STRUCT reaches its own floor by $N{=}25$ and is flat thereafter (Figure 2), so the claim concerns the regime in which data is scarce relative to the complexity of the residual.

We additionally test the benchmarks under a distribution shift on the geometry or the input function. Structural injection remains effective under this shift on five of the six, although this OOD setting does not support a general robustness claim (Appendix I).

4.3 CASE STUDY: COMPLEX GEOMETRY WITHOUT A RESIDUAL

AIRFRANS (Bonnet et al., 2022) is the benchmark on which the equation yields the least. It admits no matched residual loss, and the construction returns not a field but a single instance-side factor, the weakest of the three outputs of Eq. 9. It is where the two ways of using an equation come apart by construction. Each instance is a steady incompressible RANS solution around an airfoil: $a = (U_\infty, \alpha, s)$ gives inlet speed, incidence and 32 surface samples, and the target $u^\star \equiv p^\star$ is the pressure at $n_x = 2219$ points of a fixed 60×40 Cartesian grid that lie outside every airfoil in the dataset, referenced against finite-volume solutions on body-fitted meshes $\mathcal{M}(a)$.

A physics-informed formulation would minimize

$$\mathcal{L}_{\text{PI}}(\theta) = \sum_{i,j} \left\| \mathcal{N}^h_{a_i}[v_\theta, p_\theta, \nu_{t,\theta}](x_j) \right\|^2, \quad \mathcal{N}_a[v,p,\nu_t] = (v\cdot\nabla)v + \nabla p - \nabla\cdot\big((\nu+\nu_t)\nabla v\big). \quad (15)$$

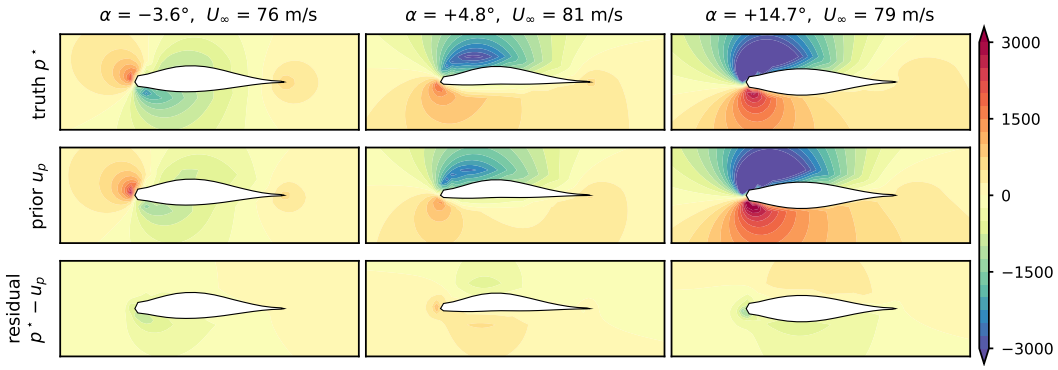

Figure 3: **What the rank-two prior removes.** Three held-out cases with $U_\infty \geq 75\,\mathrm{m\,s^{-1}}$, at the smallest, median and largest incidence among them. Top: the reference pressure $p^\star$. Middle: the prior u_p of Eq. 19, fitted on the 800-case training pool. Bottom: the residual $p^\star - u_p$ left for the corrector. All panels share one color scale.

Here v, p, ν and ν_t denote velocity, pressure, kinematic viscosity and eddy viscosity, and $v_\theta, p_\theta, \nu_{t,\theta}$ their predictions. Three obstructions meet here. The discrete operator $\mathcal{N}_a^h$ lives on $\mathcal{M}(a)$ while the x_j lie on an unaligned grid, so a stencil near the body reads points where the solution is undefined; the domain is instance-dependent, so no single residual operator is shared across the training set; and our predictor outputs pressure alone, so neither v nor ν_t is available to it; even with the true velocity, dropping ν_t leaves

$$\mathcal{N}_a[v^\star, p^\star, 0] = \nabla\cdot(\nu_t^\star \nabla v^\star) \neq 0 \tag{16}$$

at the *true* solution. None of this makes a residual loss impossible in principle: the dataset carries velocity and eddy viscosity too, and a formulation predicting them on a body-fitted discretizations could assemble one. The narrower claim is the one we rely on: under the output representation and discretizations used here, no directly matched residual formulation is available.

What survives is a scaling law rather than a stencil. Non-dimensionalizing the steady incompressible momentum equation factors the pressure exactly into a dynamic-pressure scale and a dimensionless coefficient,

$$p^\star(x;\, U_\infty, \alpha, s) = q(a)\, C_p(x;\, \alpha, s, Re), \qquad q(a) = \tfrac{1}{2}U_\infty^2, \qquad Re = \frac{U_\infty \ell}{\nu}, \tag{17}$$

in which ℓ is the chord length and the inlet speed reaches C_p only through the Reynolds number, so were C_p independent of Re the pressure would be exactly quadratic in the inlet speed. AIRFRANS holds the viscosity fixed and varies the speed, so Re ranges over 2.0–6.0 $\times$ 10^6 and dropping its argument leaves an approximation rather than an identity,

$$p^\star(x;\, \kappa U_\infty, \alpha, s) \;\approx\; \kappa^2\, p^\star(x;\, U_\infty, \alpha, s). \tag{18}$$

Its defect is confined to where molecular viscosity is not negligible against ν_t (Appendix E). The thin-airfoil linearization in α then fixes a two-dimensional instance factor and a rank-two prior,

$$\psi_{\mathcal{F}_0}(a) = q(a)\,(1,\ \alpha)^\top, \qquad u_p(x_j; a) = c(x_j)^\top \psi_{\mathcal{F}_0}(a), \tag{19}$$

with $c = (c_0, c_1)^\top$ the least-squares fit of Eq. 6 on the same N instances every formulation sees, determined for $N \geq 2$. This is the indirect prior of Eq. 9: the equation supplies the instance-side factor $\psi_{\mathcal{F}_0}$ in closed form and the labels only fill in the spatial profile.

The corrector is generic, zero-initialized and matched to the raw baseline in capacity. What matters is what the prior removes: Eq. 19 absorbs the leading-order dependence on the flow condition, leaving $r = q\big[C_p(x;\alpha,s,Re) - c_0(x) - c_1(x)\alpha\big]$, the shape-dependent departure from the rank-two field: camber and thickness, the suction peak, the wake. With u_p fitted on the 800-case training pool, $\|r\|/\|p^\star\|$ is 0.26 on the median held-out case (0.35 on average) and r is no longer tied to the flow condition: a regression on $(1, q, q\alpha)$ explains $R^2 = 0.020$ of its variance against 0.896 for $p^\star$ (Appendix E). Figure 3 shows the three fields on held-out cases. Smaller and uncoupled is what lets the same capacity buy a better fit.

Table 4 shows the consequence. PRIOR is below RAW at every budget, by 3.50$\times$ at N=10 and 1.55$\times$ at N=500, and through N=100 its error is below that of RAW at the next budget up, 0.0801

Table 4: **AIRFRANS pressure field, DeepONet backbone.** Relative L_2 error, mean $\pm$ sample standard deviation over five seeds; bold marks the column minimum, and *ratio* is RAW/PRIOR. The prior is refitted on every training draw, so $\varepsilon_{\text{prior}}$ moves with N. Paired t-tests over the shared seeds put CP-PRIOR below every other row at every budget ($p < 0.05$) and PRIOR below RAW ($p < 0.03$); PRIOR and CP-NET are resolved only at N=50 ($p = 0.03$) and N=500 ($p = 0.002$).

	training samples N					
model	10	25	50	100	200	500
RAW	1.3847 $\pm$0.7148	0.4174 $\pm$0.0511	0.2372 $\pm$0.0072	0.1473 $\pm$0.0140	0.0854 $\pm$0.0057	0.0463 $\pm$0.0034
CP-NET	1.0668 $\pm$0.6918	0.3080 $\pm$0.1069	0.1455 $\pm$0.0206	0.0782 $\pm$0.0097	0.0484 $\pm$0.0029	0.0250 $\pm$0.0014
$\varepsilon_{\text{prior}}$	0.4166	0.3788	0.3712	0.3529	0.3549	0.3545
PRIOR	0.3958 $\pm$0.0766	0.2005 $\pm$0.0246	0.1269 $\pm$0.0129	0.0801 $\pm$0.0095	0.0507 $\pm$0.0032	0.0299 $\pm$0.0007
ratio	3.50$\times$	2.08$\times$	1.87$\times$	1.84$\times$	1.69$\times$	1.55$\times$
CP-PRIOR	**0.3092** $\pm$0.1239	**0.1482** $\pm$0.0385	**0.0836** $\pm$0.0128	**0.0509** $\pm$0.0071	**0.0292** $\pm$0.0016	**0.0165** $\pm$0.0010

at N=100 against 0.0854 at N=200. The prior alone is capped, being rank two by construction, and data-derived peers overtake it between N=50 and N=100; in front of the same corrector it stays ahead of every data-derived anchor, by 3.1$\times$ at N=10 and 1.6$\times$ at N=200 (Appendices E and H). Its spatial profile is fitted rather than solved, so CP-NET separates the scaling the equation supplies from the fit: normalizing by q and learning the rest freely beats RAW by 1.76$\times$ at N=200, as much as the fitted profile (PRIOR, 1.69$\times$; the two are not resolved, $p = 0.33$). Neither substitutes for the other: CP-PRIOR keeps the anchor and multiplies the corrector of Eq. 11 by $q(a)$, $\hat{u}_\theta = u_p + q(a)\, r_\theta$, filling the instance side of Eq. 10 up to that multiplier, and is the most accurate formulation at every budget, 1.66$\times$ below CP-NET and 1.74$\times$ below PRIOR at N=200.

5 DISCUSSION

The constructions are equation specific by design. We rely on established analytical structures such as spectral expansions, transfer matrices, effective-medium reductions, complex potentials and thin airfoil theory, because they are already available from the corresponding governing equations. Our aim is not to rediscover them from data but to expose this existing knowledge before the network has to infer it from scarce labels, so that the equation supplies the structure that is already known and learning is left with what remains unresolved. The resulting constructions are specific to each equation family, but the principle behind them is shared across families. Discovering such reductions automatically is an interesting direction, though separate from the question studied here.

The gains are conditional. The prior helps roughly in proportion to how much of the solution it resolves. Where the construction leaves a large error (Darcy problems, annulus at large N), the residual loss performs comparably and occasionally better. Structural injection further requires a basis the residual actually lies in, which the first-order sensitivities on optical are not (Appendix F), and a budget large enough to estimate its coefficients, which elasticity lacks at the smallest budgets. The advantage also narrows as labels accumulate, as the AIRFRANS sweep shows. Our claims therefore concern the regime in which data are scarce relative to the complexity of the residual.

The benchmarks are small. All benchmarks are one- to three-dimensional problems at moderate resolution, and AIRFRANS, the one closest to an engineering workflow, is a steady two-dimensional field. Whether the same gains hold for unsteady three-dimensional problems at industrial resolution remains open.

6 CONCLUSION

We studied where equation knowledge should enter neural operator learning when the governing equation is known only approximately and labels are scarce, comparing its use as a training constraint, as a prior that removes what the equation already determines, and as a representation in which the remaining residual is learned. Across the benchmarks that admit a residual loss, the equation-derived prior is often more accurate than imposing the same equation as a loss, most clearly where the approximate equation resolves a large part of the solution, and AIRFRANS shows that equation knowledge remains useful even when no matched residual formulation exists. When the approximate equation also exposes the structure of the residual, learning in that basis improves accuracy further, and the gain is not explained by the added capacity. These benefits are concentrated in the regime where data are scarce relative to the complexity of the residual. Within this regime, an approximate equation is most useful when it reduces the learning problem before training, rather than when it only constrains the fit.

AI USE STATEMENT

The core research hypothesis was formulated by the authors. We used generative AI tools, including AI coding agents, to help refine the research hypotheses and develop the conceptual framework; translate author-written drafts into English; implement the numerical solvers, models, and training code, and run the experiments; clean and reformat the public datasets; give feedback on, and in some cases help design, the experiments; assist with formulating and proving mathematical claims; and interpret results. We also used them to polish the writing, draft parts of the text, create figures, and search for related work. Survey or interview design and qualitative data analysis are not applicable to this work. Data come either from public datasets or from numerical solvers; the synthetic benchmarks were generated by running AI-assisted solver code, and no data values were produced directly by a generative model. Where analytical solutions exist, we compared the solvers against them (e.g., Lamé solutions for elasticity, Poiseuille flow for the pipe prior, and single-layer closed forms for thin films), and reported results were aggregated from logged runs by scripts that exclude invalidated runs. The authors reviewed the AI-assisted code, outputs, and proofs, and checked the cited references against their sources. We take responsibility for the content of this work, including parts produced with the help of generative AI.

ETHICS STATEMENT

This work studies neural operators for partial differential equations and involves no human subjects, personal data, or sensitive attributes. All external data are publicly released scientific datasets (Li et al., 2021; 2023) and AIRFRANS (Bonnet et al., 2022), used under their respective licenses; refractive-index data are taken from a public CC0 database (`refractiveindex.info`; Appendix B.5). The remaining benchmarks are generated by numerical solvers. We are not aware of conflicts of interest or of direct harmful applications beyond those common to surrogate modeling in engineering.

REPRODUCIBILITY STATEMENT

The equation-derived constructions for every benchmark, including solver details, are given in Appendix B, and the factorization of the structured branch in Appendix B.8. Benchmark specifications are given in Appendix C, and splits, architectures, parameter counts, training settings and budgets in Appendix C.2, with timing in Appendix L; the selection of the residual-loss weight is described in Appendix C.1, and the AIRFRANS pipeline in Appendix E. All results are averaged over five fixed seeds. Complete proofs of the identities used in Section 2 are in Appendix A. Source code for data generation, training and table aggregation will be released upon publication.

REFERENCES

Maxime Barrault, Yvon Maday, Ngoc Cuong Nguyen, and Anthony T. Patera. An 'empirical interpolation' method: application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathématique*, 339(9):667–672, 2004.

Florent Bonnet, Jocelyn Ahmed Mazari, Paola Cinnella, and Patrick Gallinari. AirfRANS: High fidelity computational fluid dynamics dataset for approximating Reynolds-averaged Navier–Stokes solutions. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, volume 35, pp. 23463–23478, 2022.

Jennỳ Brynjarsdóttir and Anthony O'Hagan. Learning about physical parameters: The importance of model discrepancy. *Inverse Problems*, 30(11):114007, 2014.

Saifon Chaturantabut and Danny C. Sorensen. Nonlinear model reduction via discrete empirical interpolation. *SIAM Journal on Scientific Computing*, 32(5):2737–2764, 2010.

Nanxi Chen, Chuanjie Cui, Airong Chen, Sifan Wang, and Rujin Ma. On the training of physics-informed neural operators for solving parametric partial differential equations. *arXiv preprint arXiv:2606.06164*, 2026.

Ha Dang, Sebastian Schmidt, and Juergen Hesser. Smooth piecewise cutting for neural operator to handle discontinuities and sharp transitions. *arXiv preprint arXiv:2605.19823*, 2026.

Alexander I. J. Forrester and Andy J. Keane. Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences*, 45(1-3):50–79, 2009.

Zhongkai Hao, Zhengyi Wang, Hang Su, Chengyang Ying, Yinpeng Dong, Songming Liu, Ze Cheng, Jian Song, and Jun Zhu. GNOT: A general neural operator transformer for operator learning. In *International Conference on Machine Learning*, pp. 12556–12569, 2023.

Marc C Kennedy and Anthony O'Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(3):425–464, 2001.

Dmitrii Kochkov, Jamie A. Smith, Ayya Alieva, Qing Wang, Michael P. Brenner, and Stephan Hoyer. Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, 118(21):e2101784118, 2021.

Jae Yong Lee, Sung Woong Cho, and Hyung Ju Hwang. HyperDeepONet: Learning operator with complex target function space using the limited resources via hypernetwork. In *International Conference on Learning Representations*, 2023.

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

Zongyi Li, Daniel Zhengyu Huang, Burigede Liu, and Anima Anandkumar. Fourier neural operator with learned deformations for PDEs on general geometries. *Journal of Machine Learning Research*, 24(388):1–26, 2023.

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. *ACM/IMS Journal of Data Science*, 1(3):1–27, 2024.

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.

Lu Lu, Xuhui Meng, Shengze Cai, Zhiping Mao, Somdatta Goswami, Zhongqiang Zhang, and George Em Karniadakis. A comprehensive and fair comparison of two neural operators (with practical extensions) based on FAIR data. *Computer Methods in Applied Mechanics and Engineering*, 393:114778, 2022.

Lei Ma, Nicolas Boullé, Yu-Sen Yang, Hao Wu, and Ling Guo. Physics-guided correction for operator learning under model misspecification. *arXiv preprint arXiv:2606.03469*, 2026a.

Siying Sydney Ma, Mehrdad Momeni Zadeh, Mauricio Soroco, Wuyang Chen, Jiguo Cao, and Vijay Ganesh. Learning data-efficient and generalizable neural operators via fundamental physics knowledge. In *International Conference on Learning Representations*, 2026b.

Xuhui Meng and George Em Karniadakis. A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse PDE problems. *Journal of Computational Physics*, 401:109020, 2020.

Apostolos F. Psaros, Xuhui Meng, Zongren Zou, Ling Guo, and George Em Karniadakis. Uncertainty quantification in scientific machine learning: Methods, metrics, and comparisons. *Journal of Computational Physics*, 477:111902, 2023.

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

Ehsan Roohi and Amirmehran Mahdavi. Shock-centered low-rank structure and neural-operator representation of rarefied micro-nozzle flows. *arXiv preprint arXiv:2605.12723*, 2026.

Lawrence Sirovich. Turbulence and the dynamics of coherent structures. Part I: Coherent structures. *Quarterly of Applied Mathematics*, 45(3):561–571, 1987.

Kiwon Um, Robert Brand, Yun Raymond Fei, Philipp Holl, and Nils Thuerey. Solver-in-the-loop: Learning from differentiable physics to interact with iterative PDE-solvers. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6111–6122, 2020.

Sifan Wang, Hanwen Wang, and Paris Perdikaris. Learning the solution operator of parametric partial differential equations with physics-informed DeepONets. *Science Advances*, 7(40):eabi8605, 2021.

Sifan Wang, Jacob H. Seidman, Shyam Sankaran, Hanwen Wang, George J. Pappas, and Paris Perdikaris. CViT: Continuous vision transformer for operator learning. In *International Conference on Learning Representations*, 2025.

Xihang Yue, Yi Yang, and Linchao Zhu. DeltaPhi: Physical states residual learning for neural operators in data-limited PDE solving. In *Advances in Neural Information Processing Systems*, 2025.

A Two Exact Identities Behind the First Two Slots

This appendix proves Lemma 1 and Proposition 1 of Section 2. Both are exact algebraic consequences of Eq. 3, and neither assumes $\|\Delta\|$ small. Both are stated for a linear L_0. The lemma further assumes that Δ acts in the interior and that u and u_p share the boundary condition, which holds on 2D Poisson, 1D heat, battery, the Darcy and Burgers benchmarks and Pipe; on 3D layered and annulus Δ acts on interfaces, on elasticity the boundary itself differs, and Section 2 states what replaces the lemma there. Neither assumes Δ linear: $\Delta(u)$ is evaluated at the truth, so the lemma is exact on the two Burgers benchmarks (heat-equation L_0) and on Pipe (Stokes L_0), whose reference equations are nonlinear. Where $\mathcal{F}_0$ is itself nonlinear (1D nonlinear reaction, AirfRANS) both are a linearization rather than an identity, and we do not apply them there.

Proof of Lemma 1. Write $\mathcal{F}(v) = L_0 v - f + \Delta(v)$, with L_0 boundedly invertible on the space of fields satisfying the homogeneous form of the shared boundary operator $\mathcal{B}$. Subtracting $L_0 u_p = f$ from $L_0 u + \Delta(u) = f$ gives $L_0(u - u_p) = -\Delta(u)$. Both u and u_p satisfy $\mathcal{B}$, so r lies where L_0 is invertible; apply L_0^{-1}. □

If L_0 is self-adjoint with eigenpairs (ϕ_m, κ_m) and $\Delta(u) = \sum_m c_m \phi_m$, then $r = -\sum_m (c_m/\kappa_m)\phi_m$, so when L_0 admits a useful eigenbasis the same equation-derived basis provides a natural coordinate system for the residual. We do not claim a general decay rate for c_m across benchmarks; any observed decay is treated as an empirical property of the corresponding benchmark construction rather than a universal law.

Section 2 places the residual loss in the objective slot of Eq. 10. For a linear L_0 the penalized objective is quadratic in $\hat{u}$, with Hessian $2M$, $M = I + \lambda L_0^\top L_0$, and minimizer v_λ, so completing the square gives

$$\|\hat{u} - u\|^2 + \lambda \|L_0 \hat{u} - f\|^2 = \|\hat{u} - v_\lambda\|_M^2 + c, \tag{20}$$

with c independent of $\hat{u}$. The loss therefore fits $\hat{u}$ to v_λ in the metric M: it reweights the error, and it moves the target unless $\Delta(u) = 0$, where $v_\lambda = u$.

Proof of Proposition 1. Stationarity gives $\left(I + \lambda L_0^\top L_0\right) v = u + \lambda L_0^\top f$. Substituting $f = L_0 u + \Delta(u)$, which is Eq. 3 at the truth, makes the right side $\left(I + \lambda L_0^\top L_0\right) u + \lambda L_0^\top \Delta(u)$; multiply by $\left(I + \lambda L_0^\top L_0\right)^{-1}$. □

The displacement from u depends on λ and $\Delta(u)$ alone, vanishes when $\Delta(u) = 0$, and carries no dependence on the sample size N. This is a statement about the exact population minimizer under unlimited optimization: a finite-sample, finite-capacity, finite-step estimator need not reach v_λ, so the claim is that the penalty's *target* does not sharpen as labels accumulate, not that every trained instance of it will exhibit an unchanging gap.

B Equation-Derived Constructions

This appendix specifies the equation-derived constructions used throughout the experiments. The main text represents all such constructions abstractly through u_p and $\Phi_{\mathcal{F}_0}$; here we give the concrete forms used for each problem class. Table 5 names the construction each benchmark uses and what, if anything, it exposes beyond a field; the subsections that follow derive them.

Table 5: **Prior construction for every benchmark.** Every construction is solved independently of the solution labels, AIRFRANS excepted as noted. "None" means the solve returns a field and nothing reusable beyond it.

Construction	Benchmarks	Prior is	Structure exposed
Direct solve of $\mathcal{F}_0$	1D heat, battery, nonlinear reaction	a numerical solution	none
Reduced equation	1D convection	forward integration of $Vu' = f$	none
Heat equation	1D Burgers	$u_t = \nu u_{xx}$ at $t = 0.1$, nonlinearity dropped	none
	Burgers (Li et al., 2021)	$u_t = \nu u_{xx}$ at $t = 1$, nonlinearity dropped	none
Effective medium	1D Darcy	harmonic-mean coefficient	none
	2D Darcy	geometric-mean coefficient	none
	Darcy (Li et al., 2021)	geometric-mean coefficient	none
	3D Darcy	geometric-mean coefficient	none
Stokes solve	Pipe (Li et al., 2023)	steady Stokes flow in the same	Green response of $\mathcal{F}_0$
	Pipe-GRF	geometry, inertia dropped	none
Spectral expansion	2D Poisson	eigenfunction sum	modal basis
Transfer matrix	3D layered (four layers)	layer-wise closed form	modal basis, interfaces $b_j(a)$
Characteristic matrix	optical thin film	2×2 product per wavelength	sensitivity directions
Polar transfer matrix	annulus	mode-wise closed form	radial modes
Complex potential	elasticity	conformal-map solution	Laurent modes
Thin airfoil	AirfRANS	$q(c_0 + c_1\alpha)$, two fitted	none

B.1 Spectral Representation

For problems on separable domains, the linear operator L_0 of $\mathcal{F}_0(u; a) = L_0 u - f$ admits orthonormal eigenpairs (ϕ_m, κ_m) satisfying

$$L_0\phi_m = \kappa_m\phi_m.$$

Expanding the solution in this basis gives the equation-derived prior

$$u_p(x; a) = \sum_{m \le m_{\max}} \frac{\langle f(\cdot\,; a), \phi_m \rangle}{\kappa_m} \phi_m(x). \tag{21}$$

The same eigenfunctions, multiplied by the source coefficients $\langle f, \phi_m \rangle$, form the structured branch; Appendix B.8 gives this factorization as form (a).

Verification. Where a closed form exists we check against it: the spectral prior reaches 2.4×10^{-7} against $x(1-x)$ for one-dimensional Poisson, and is exact to machine precision for the heat equation with a single mode source and for two-dimensional Poisson with a product source. These verify the construction, not the accuracy of any prior as a predictor; how far u_p is from the truth is a property of how incomplete $\mathcal{F}_0$ is and is reported in the main tables.

B.2 Layered Transfer Construction

For layered problems, the solve separates the transverse coordinate y from the layer-normal coordinate z, writing the query coordinate as $x = (y, z)$. Let $\{\phi_m(y)\}$ denote the transverse modes and let w_i denote the layer widths. The interface locations are

$$b_j(a) = \sum_{i \le j} w_i. \tag{22}$$

The four layer conductivities and the interface resistance are fixed across instances, so only the source and the widths vary. The transfer construction solves each transverse mode in closed form inside every layer and matches the layers at these interfaces; the prior imposes continuity of the field and of the flux there, and the reference adds an interface thermal resistance. The transverse modes and the layer-wise z-solutions form the separable spatial factor of form (b) in Appendix B.8.

B.3 Polar Transfer Construction

For disk and annulus geometries, we use the corresponding polar form of the transfer construction. Cartesian layer solutions are replaced by radial solutions of the separated equation, while interface matching is retained in the same transfer form. This allows curved interfaces to be handled through the radial coordinate without changing the surrounding operator-learning procedure.

Writing the query coordinate in polar form $x = (\rho, \vartheta)$, each angular mode m has the radial solutions ρ^m and ρ^{-m} in every region, matched at the interface radii; the prior imposes continuity of the field and of the flux there, and the reference adds an interface thermal resistance. The angular modes and these region-wise radial solutions form the separable spatial factor of form (b) in Appendix B.8.

B.4 Plane Elasticity

For plane elasticity, the equation-derived solve is formulated through complex potentials, with the exterior of the hole mapped conformally onto the exterior of the unit disk. The prior solves the infinite plate under the prescribed hole traction; the reference solves a finite plate with a traction-free outer boundary under the same traction. The stress fields of the Laurent expansion through the conformal map form the spatial factor of form (c) in Appendix B.8, shared by every Fourier coefficient of the traction.

B.5 Optical Thin Films

For the optical thin-film problem, the field at each wavelength is propagated through the multilayer stack by a product of 2×2 characteristic matrices, one per layer. The phase accumulated in layer j is

$$\delta_j(\lambda) = \frac{2\pi n_j(\lambda) d_j}{\lambda}, \tag{23}$$

where λ is the wavelength (unrelated to the loss weight of Eq. 4, which is not defined on this benchmark) and n_j and d_j are the refractive index and thickness of layer j. The equation-derived prior uses constant refractive indices and neglects absorption, whereas the reference solution uses tabulated dispersive $n_j(\lambda)$ and extinction coefficients $k_j(\lambda)$, taken from the public CC0 database `refractiveindex.info`. The resulting residual therefore reflects material dispersion and absorption rather than an omitted differential term.

The characteristic-matrix solve also exposes sensitivity directions: the derivatives of the prior reflectance with respect to the composition in each depth bin, which form the spatial factor of form (d) in Appendix B.8.

B.6 Stokes Construction for Pipe Flow

For the pipe benchmarks the reference equation is the steady incompressible Navier–Stokes system at $\nu = 5 \times 10^{-3}$ in a curved channel of length 10 and unit vertical width, and $\mathcal{F}_0$ drops the inertial term, leaving steady Stokes flow in the same geometry under the same boundary conditions. Both are solved by the same stream-function–vorticity discretizations, so the prior and the reference differ by the convective term alone: the missing physics is inertia, at $\mathrm{Re} \approx 33$.

Writing $y_b(x)$ for the lower wall, the solve is carried out in the computational coordinates $\xi = x$, $\eta = y - y_b(x) \in [0, 1]$, in which every instance shares one grid. With $s = y_b'$ the Laplacian becomes

$$\nabla^2_{a_g} f = f_{\xi\xi} - 2s\, f_{\xi\eta} + (1 + s^2) f_{\eta\eta} - s' f_\eta, \tag{24}$$

and the prior solves

$$\nabla^2_{a_g} \psi + \omega = 0, \qquad \nabla^2_{a_g} \omega = 0, \qquad u_p = \partial_\eta \psi, \tag{25}$$

where the reference instead carries $\nabla^2_{a_g} \omega = \nu^{-1} \big[u\, \omega_\xi + (v - su)\, \omega_\eta \big]$ on the right. The boundary conditions are no-slip walls, $\psi = 0$ and $\psi = \frac{1}{6}$ with Thom's condition $\omega_w = -(1 + s^2)\, 2(\psi_1 - \psi_w)/\Delta\eta^2$; a Poiseuille inlet, $\psi = \eta^2/2 - \eta^3/3$ and $\omega = -(1 - 2\eta)$; and second-order extrapolation at the outlet. Both equations are discretized by centered differences on the same 129×129 grid, so the prior is one sparse solve per instance.

Verification. On a straight channel the construction reproduces the Poiseuille profile to $\max |u_p - \eta(1 - \eta)| = 6 \times 10^{-5}$. Against the reference labels it gives $\varepsilon_{\mathrm{prior}} = 0.107$ on Pipe and 0.084 on Pipe-GRF. Lubrication theory is not used as a prior: in these coordinates $\eta(1 - \eta)$ is a shape-independent constant field, so its error coincides with that of the training mean field (0.162 against 0.160), and a prior that ignores the geometry would not test the construction.

Green response. Because the geometry enters the operator of Eq. 25 rather than its data, the construction exposes a representation through the same operator rather than through a basis of the source. Let $\mathcal{S}_{a_g}$ denote the discrete Stokes operator just described, with its boundary rows made homogeneous, and let $\chi_k(\xi,\eta) = \sin(m\pi\xi/10)\sin(n\pi\eta)$ be a probe source entered in the interior rows of the vorticity equation only. The k-th column of $\Phi_{\mathcal{F}_0}$ is the horizontal velocity of the response,

$$S_k(x;a_g) = \sigma_k^{-1}\,\partial_\eta\psi_k, \qquad \nabla^2_{a_g}\psi_k + \omega_k = 0, \quad \nabla^2_{a_g}\omega_k = \chi_k, \tag{26}$$

with $m \le 32$ and $n \le 4$, giving $n_b = 128$, and σ_k the root-mean-square of the same column computed once on the straight channel $y_b \equiv 0$, a fixed reference shared by every shape. All 128 columns share the coefficient matrix of the prior solve, so they cost one factorization per shape.

The geometry input is $a_g = c(y_b) \in \mathbb{R}^9$, the pool principal-component coordinates of the lower wall divided by their pool standard deviations; the labels are not used. Nine coordinates are exact rather than truncated, since the benchmark's shape family is a ten-parameter spline and y_b lies in a nine-dimensional affine subspace to 8×10^{-9} of the leading singular value. There is no source factor: the instance is the geometry alone, so $F_k \equiv 1$ and $\Phi_{\mathcal{F}_0}$ does not separate into an instance factor times a spatial one. This is form (e) in Appendix B.8, and it is why the matched controls on this benchmark replace the whole of $\Phi_{\mathcal{F}_0}$ by a sample-independent basis rather than replacing S alone.

The width is chosen by the same least-squares diagnostic used elsewhere: with $\beta^\star = \arg\min_\beta \|u - u_p - \Phi_{\mathcal{F}_0}\beta\|$ per sample, $n_b = 128$ leaves 8.6% of the residual outside the span on average (median 4.3%) and an oracle error of 0.011, against 19.8% and 0.023 at $n_b = 64$; fitting $a_g \mapsto \beta^\star$ by kernel ridge from 200 training shapes reaches 0.023 at $n_b = 128$ and 0.030 at $n_b = 64$, against 0.107 for u_p alone and 0.107 for a random basis of the same width.

Pipe-GRF replaces the spline shape family by a Gaussian random field restricted to the statistics of the original family, and uses the same prior construction; no $\Phi_{\mathcal{F}_0}$ is built for it.

B.7 Effective-Medium Darcy Construction

For Darcy problems with rough coefficients, the effective-medium construction replaces the spatially varying coefficient $a(x)$, which is itself the instance input a, by a constant effective value a_0 and solves the resulting constant-coefficient problem exactly. In one dimension, the effective coefficient is the harmonic mean,

$$a_0 = \left\langle a^{-1} \right\rangle^{-1}, \tag{27}$$

where $\langle\cdot\rangle$ denotes the spatial average over Ω; this is the exact effective coefficient for the one-dimensional flux problem. For the two- and three-dimensional isotropic cases, we use the standard geometric-mean effective value

$$a_0 = \exp(\langle \log a \rangle)\,. \tag{28}$$

These constructions provide an equation-derived prior u_p but do not expose a reusable representation for the discarded coefficient variation. They therefore terminate at Level 2 and use a generic residual neural operator rather than structural injection.

B.8 Factorization of the Structured Branch

This subsection gives, for each benchmark with a structured branch, the source factor F, the geometry input a_g and the spatial factor S of Eq. 14 (Table 6). The spatial factor takes one of five forms.

Table 6: **Factors of the structured branch.** Source factor F, geometry input a_g and spatial factor S of Eq. 14, with the width n_b of $\Phi_{\mathcal{F}_0}$.

Form	Benchmark	$F_k(a_f)$	a_g	$S_k(x;a_g)$	β_θ	n_b
(a) fixed	2D Poisson	$\hat{f}_m$	–	$\phi_m(x)$	constant	256
(b) separable	annulus	$\hat{g}^s_m$	$(r_j)_{j\le4}$	$P_{ms}(\vartheta)\,Z_{ml}(\rho;a_g)$	$\beta_\theta(a_g)$	216
	3D layered	$c_m\eta_{k'}$	$(w_j)_{j\le4}$	$\phi_m(y)\,Z_{mj\sigma}(z;a_g)$	$\beta_\theta(a_g)$	3072
(c) shared	elasticity	$\hat{t}_j$	(m_1,m_2,m_3)	$B_k(x;a_g)$	$\beta_\theta(a_g)$	512
(d) sensitivity	optical	1	$\theta(z)$	$\partial R_p/\partial\theta_k(\lambda;a_g)$	constant	64
(e) response	Pipe	1	$c(y_b)\in\mathbb{R}^9$	$\sigma_k^{-1}\partial_\eta\psi_k(x;a_g)$	$\beta_\theta(a_g)$	128

(a) Fixed basis. When the domain does not vary, a_g is empty and $S_k(x)$ is a fixed eigenbasis. For 2D Poisson on $[0,1]^2$, with $m=(p,q)$ and $1 \le p,q \le 16$,

$$F_m(a_f) = \hat{f}_m, \qquad S_m(x) = \phi_m(x) = 2\sin(p\pi x_1)\sin(q\pi x_2), \tag{29}$$

where $\hat{f}_m$ are the sine coefficients of the source. With the reference equation $-\nabla^2 u + \mu u = f$ and the prior $-\nabla^2 u_p = f$, the residual inside the span is $\sum_m \hat{f}_m\,[(\kappa_m+\mu)^{-1} - \kappa_m^{-1}]\,\phi_m(x)$ with $\kappa_m = (p^2+q^2)\pi^2$, so the target coefficients $\beta^\star_m = (\kappa_m+\mu)^{-1} - \kappa_m^{-1}$ do not depend on the instance.

(b) Separable basis. When the domain is a product $x = (x_\parallel, x_\perp)$ whose geometry varies only across $x_\perp$, the index splits as $k=(m,l)$ and

$$S_{ml}(x; a_g) = P_m(x_\parallel)\, Z_{ml}(x_\perp; a_g), \tag{30}$$

a fixed factor P_m along the directions in which the geometry is invariant times a geometry-dependent factor Z_{ml} across them. For the annulus, $x=(\rho,\vartheta)$, a_f is the boundary data $g(\vartheta)$ with Fourier coefficients $\hat{g}^s_m$, and $a_g = (r_1,\dots,r_4)$ are the interface radii, with $r_0=0$ and $r_5=1$. With $m \le 12$ and $s \in \{\cos,\sin\}$,

$$F_{msl}(a_f) = \hat{g}^s_m, \qquad P_{ms}(\vartheta) = s(m\vartheta), \qquad Z_{ml}(\rho; a_g) \in \Big\{ \big(\tfrac{\rho}{r_{j+1}}\big)^m \chi_j(\rho),\ \big(\tfrac{r_j}{\rho}\big)^m \chi_j(\rho) \Big\}_{j=0}^{4}, \tag{31}$$

where χ_j is the indicator of $[r_j, r_{j+1})$ and region $j=0$ keeps only the first function, giving nine radial functions and $n_b = 2\cdot 12\cdot 9 = 216$. For the 3D layered medium, $x=(y,z)$ with $y \in [0,1]^2$, the source is $f(y,z) = \sum_{m,k'} c_m\,\eta_{k'}\,\phi_m(y)\sin(k'\pi z)$, and $a_g = (w_1,\dots,w_4)$ are the layer widths with interfaces b_j of Eq. 22. With $m=(p,q)$, $p,q \le 8$, $k' \le 4$, layer $j \in \{1,\dots,4\}$ and $\sigma \in \{0,+,-\}$,

$$F_{mk'j\sigma}(a_f) = c_m\,\eta_{k'}, \quad P_m(y) = \phi_m(y), \quad Z_{mj\sigma}(z; a_g) = \chi_j(z) \times \begin{cases} 1, & \sigma = 0, \\ e^{-\gamma_m (z - b_{j-1})}, & \sigma = +, \\ e^{-\gamma_m (b_j - z)}, & \sigma = -, \end{cases} \tag{32}$$

where $\phi_m(y) = \sin(p\pi y_1)\sin(q\pi y_2)$, $\gamma_m = \pi\sqrt{p^2+q^2}$ and χ_j is the indicator of $[b_{j-1}, b_j)$. F does not depend on (j,σ) and Z does not depend on k', giving $n_b = 64\cdot 4\cdot 12 = 3072$: 64 lateral modes, 4 source modes in z, and 3 functions in each of the 4 layers.

(c) Shared fields. When the equation supplies the fields but not the geometry-dependent map from a_f to their coefficients, each field is paired with every coefficient of a_f. Indexing the branch by the pair (k,j),

$$\begin{gathered} F_{kj}(a_f) = \hat{t}_j, \qquad S_{kj}(x; a_g) = B_k(x; a_g), \\ \hat{u}_\theta(x; a) = u_p(x; a) + \sum_k \Big[\sum_j \beta_{\theta,kj}(a_g)\,\hat{t}_j \Big] B_k(x; a_g) + \epsilon_\theta(x; a). \end{gathered} \tag{33}$$

For plane elasticity, x is a query point together with a stress component, a_f is the hole traction $\tilde{t}(\vartheta)$ with Fourier coefficients $\hat{t}_j$ (two components, cosine and sine, modes 1 and 2; 8 in total), $a_g = (m_1, m_2, m_3)$ are the conformal-map coefficients, and B_k are the 64 stress fields of the Laurent expansion through that map with orders $|k| \le 8$, giving $n_b = 64\cdot 8 = 512$. At fixed shape the residual is linear in the load, $u - u_p = \sum_{k,j} W_{kj}(a_g)\,\hat{t}_j\,B_k(x; a_g) + \epsilon$, so β_θ learns the map $W(a_g)$.

(d) Sensitivity basis. When the residual has no linear instance data, $F_k \equiv 1$ and the spatial factor is a derivative of the prior. For optical thin films, $x=\lambda$, the instance is the TiO_2 fraction profile $a=\theta(z)$, and with θ_k its value in depth bin $k \le 64$,

$$S_k(\lambda; a) = \frac{\partial R_p}{\partial \theta_k}(\lambda; a), \tag{34}$$

the Fréchet derivative of the prior reflectance, evaluated by central differences, with β_θ a constant vector.

(e) Operator response. When the geometry enters the operator itself, no factor of $\Phi_{\mathcal{F}_0}$ is free of it and $F_k \equiv 1$: the spatial factor is the prior operator's own response to a fixed probe source, Eq. 26.

Table 7: **The eighteen benchmarks and the level of Eq. 1 each one exercises.** "Prior" is *direct* when $\mathcal{S}(\mathcal{F}_0, a)$ returns the field itself and no solution label is consumed (Eq. 5), *indirect* when the equation returns an instance-side factor whose spatial coefficients are fitted by least squares (Eq. 6). "Structure" names $\Phi_{\mathcal{F}_0}$ where the construction exposes one; "–" means it exposes none. The PI column marks the eleven benchmarks admitting a differentiable residual at all. FNO applies only where a is a field over the domain; elsewhere the backbone is DeepONet (DON). Benchmarks with a structured branch use only DeepONet, the backbone extended by PRIOR-STRUCT except for Pipe; 2D Poisson satisfies the FNO condition but is included among these benchmarks. Why the residual loss is absent: [a] the output is a single time slice, so u_t cannot be formed; [b] equilibrium holds for the prior, the reference and the zero field alike, so the residual can be assembled but does not identify the solution, which only the boundary data select (Appendix C.2); [c] the reference comes from tabulated dispersive optical constants, so the discrepancy is measured material data rather than an omitted differential term; [d] the target is one velocity component, so the momentum residual would need the pressure and the transverse component; [e] the discrete operator lives on an instance-dependent mesh and the RANS system is not closed (Section 4.3). Specifications: Table 9; constructions: Table 5.

Benchmark	Construction	Prior	Structure $\Phi_{\mathcal{F}_0}$	PI	Backbones
Eleven benchmarks admitting a differentiable residual: RAW, PI *and* PRIOR *are matched*					
1D nonlin. reaction	direct solve	direct	–	✓	DON, FNO
1D convection	reduced equation	direct	–	✓	DON, FNO
1D heat	direct solve	direct	–	✓	DON, FNO
battery	direct solve	direct	–	✓	DON, FNO
1D Darcy	effective medium	direct	–	✓	DON, FNO
2D Darcy	effective medium	direct	–	✓	DON, FNO
3D Darcy	effective medium	direct	–	✓	DON, FNO
Darcy (Li et al., 2021)	effective medium	direct	–	✓	DON, FNO
2D Poisson	spectral expansion	direct	modal basis	✓	DON
3D layered	transfer matrix	direct	modal basis, interfaces	✓	DON
annulus	polar transfer matrix	direct	radial modes	✓	DON
Seven benchmarks admitting no residual loss: the equation enters as a prior or not at all					
1D Burgers	heat equation	direct	–	–[a]	DON, FNO
Burgers (Li et al., 2021)	heat equation	direct	–	–[a]	DON, FNO
elasticity	complex potential	direct	Laurent modes	–[b]	DON
optical	characteristic matrix	direct	sensitivity directions	–[c]	DON
Pipe (Li et al., 2023)	Stokes solve	direct	Stokes Green response	–[d]	DON, FNO
Pipe-GRF	Stokes solve	direct	–	–[d]	DON, FNO
AIRFRANS	thin airfoil	indirect	–	–[e]	DON

For Pipe, x is a point of the computational grid, a_g the nine shape coordinates, and S_k the horizontal velocity of $\mathcal{S}_{a_g}^{-1}\chi_k$ for $\chi_k = \sin(m\pi\xi/10)\sin(n\pi\eta)$ with $m \le 32$, $n \le 4$, giving $n_b = 128$. Unlike form (d) the coefficients are not constant: the operator moves with the instance, so β_θ must read the shape.

C EXPERIMENT 1: SETTINGS AND RAW RESULTS

Table 7 is the roadmap for this appendix: it names, for every benchmark, the construction used, whether the resulting prior is direct or indirect, whether the construction exposes a reusable residual representation, and whether a residual loss exists at all. Table 9 then gives the numerical specifications and Table 5 the constructions themselves.

C.1 HOW THE RESIDUAL WEIGHT IS SELECTED

The weight λ of PI is chosen by lowest error on the held-out set, and that held-out set is the one on which the reported error is computed. The selection is therefore an oracle: PI is handed a λ chosen with knowledge of the evaluation, which no honest procedure could supply. This is deliberate, so that the loss is compared at its best rather than at a weight we happened to pick, but it invites the question of whether the prior's advantage survives a fair selection rule.

It does, and for a structural reason. The oracle returns the minimum over the grid, so it upper-bounds every selection rule defined on that grid: any rule that does not see the evaluation can only match it or do worse. The reported margin of PRIOR over PI is thus a lower bound on the margin against

Table 8: **How the comparison depends on the rule that picks λ.** Each row counts the runs (cell $\times$ seed, a cell being benchmark $\times$ backbone $\times$ budget) in which PRIOR is the more accurate of the two, over all nineteen benchmark–backbone pairs admitting PI, the three linear benchmarks included (93 cells, 465 runs). The oracle is the rule used throughout the paper and the most favorable to PI on this grid.

Rule for choosing λ	PRIOR better
Oracle: lowest error on the evaluation set (used throughout)	$349/465$
Leave-one-seed-out: chosen on the other four seeds	$360/465$
Mean over the nine-point grid	$437/465$
Worst λ on the grid	$455/465$

Table 9: **Numerical specification of every benchmark.** m is the input dimension the branch sees, d_q the query dimension, n_q the number of query points, n_b the width of $\Phi_{\mathcal{F}_0}$ where one exists, and n_{test} the size of the shared held-out set. "prior" is $\varepsilon_{\text{prior}}$, the relative L_2 error of u_p alone on that set; on AIRFRANS it is the value at $N = 200$, the prior being refitted on every draw. Group A exposes no residual representation and group B does. The FNO column marks the benchmarks run under FNO: those whose input is a field over the domain, except group B, which is run under DeepONet only apart from Pipe.

Benchmark	gr.	m	d_q	n_q	n_b	n_{test}	prior	FNO	Φ
1D nonlin. reaction	A	128	1	201	–	150	0.0029	✓	–
1D convection	A	128	1	201	–	150	0.1528	✓	–
1D heat	A	128	1	201	–	150	0.4930	✓	–
battery	A	128	1	201	–	150	0.5220	✓	–
1D Darcy	A	128	1	201	–	150	0.2245	✓	–
1D Burgers	A	128	1	201	–	150	0.2069	✓	–
2D Darcy	A	4096	2	4096	–	150	0.1964	✓	–
3D Darcy	A	32768	3	32768	–	100	0.1895	✓	–
Darcy (Li et al., 2021)	A	7225	2	7225	–	200	0.2511	✓	–
Burgers (Li et al., 2021)	A	1024	1	1024	–	200	0.3548	✓	–
2D Poisson	B	4096	2	4096	256	150	0.1827	–	✓
3D layered	B	1156	3	8192	3072	150	0.1751	–	✓
annulus	B	132	2	1024	216	150	0.6649	–	✓
elasticity	B	259	3	4608	512	150	0.3214	–	✓
optical	B	128	1	512	64	150	0.2941	–	✓
Pipe (Li et al., 2023)	B	16641	2	16641	128	200	0.1070	✓	✓
Pipe-GRF	A	16641	2	16641	–	200	0.0842	✓	–
AIRFRANS	A	34	2	2219	–	200	0.3549	–	–

an honestly tuned loss, and the direction of the comparison cannot be reversed by improving the selection on this grid. Where the oracle sits at its upper edge (λ^*=1 on 3D layered, on 1D nonlinear reaction under FNO at $N \leq 50$, and on 1D convection under FNO at N=10), the prior's margin is large enough (1.0350 against 0.2377 at N=200; 0.3188 against 0.0030 and 0.2759 against 0.0770 at N=10) that a wider grid would not change the conclusion.

Table 8 confirms that the bound is not vacuous. Alongside the oracle we score three alternatives on the same runs: a leave-one-seed-out rule, in which λ is chosen by the mean over the other four seeds and evaluated on the seed held out, so the evaluated run never enters the selection; the mean over the nine-point grid, the expected result of choosing λ at random; and the worst λ on the grid. Moving from the oracle to the honest rule raises the count in favour of the prior from 349 to 360 of 465 runs, and the two weaker rules raise it further, to 437 and 455. Across all 4185 combinations of a matched run and a grid value, PI is below PRIOR in 676; at $N = 200$ the cells that admit any such λ are 1D Darcy under both backbones and 2D Darcy, 3D Darcy, Darcy (Li et al., 2021), 1D heat and battery under FNO, that is, the Darcy family and two of the linear benchmarks excluded from Table 13; at $N \geq 500$ annulus under DeepONet joins them.

C.2 EXPERIMENTAL SETTING

Operators. We take both backbones and their training protocol from the reference implementations rather than tuning them. *DeepONet* is the DEEPONETCP of the PINO release: branch MLP $[m] + [50] \times 4$ and trunk MLP $[d_q] + [50] \times 4$, ReLU, no activation on the last layer of either, and no bias term b_0, so it differs from Eq. 2 of Lu et al. (2021) in those two respects. Inputs are z-scored on the training instances and the output is normalized by a single scalar mean and standard deviation, the

Table 10: **Trainable parameters, counted from the instantiated models.** RAW, PI and PRIOR are the same network and therefore the same count. PRIOR-STRUCT, PRIOR-RAND and PRIOR-POD share a count, which is what makes the last two capacity-matched controls. Counts with the generic corrector deleted: Table 17. A dash marks a backbone or formulation the benchmark does not admit. Complex spectral weights of FNO count once; counting real and imaginary parts separately gives 549,569, 2,368,001 and 6,558,197 for the 1D, 2D and 3D FNO.

Benchmark	DeepONet	FNO	PRIOR-STRUCT	PRIOR-RAND / PRIOR-POD
	RAW = PI = PRIOR			
1D nonlin. reaction	21,850	287,425	–	–
1D convection	21,850	287,425	–	–
1D heat	21,850	287,425	–	–
battery	21,850	287,425	–	–
1D Darcy	21,850	287,425	–	–
1D Burgers	21,850	287,425	–	–
2D Darcy	220,300	1,188,353	–	–
3D Darcy	1,653,950	3,281,397	–	–
Darcy (Li et al., 2021)	376,750	1,188,353	–	–
Burgers (Li et al., 2021)	66,650	287,425	–	–
2D Poisson	220,300	–	220,558	220,558
3D layered	73,350	–	235,374	235,374
annulus	22,100	–	38,468	38,468
elasticity	28,500	–	59,914	59,914
optical	21,850	–	21,916	21,916
Pipe (Li et al., 2023)	847,550	1,188,353	859,680	859,680
Pipe-GRF	847,550	1,188,353	–	–
AIRFRANS	17,200	–	–	–

convention the FNO reference uses. *FNO* (Li et al., 2021) has four spectral blocks $x \mapsto \mathcal{K}(x) + Wx$ with GELU on the first three, a head $\mathrm{fc}_1 \to 128 \to \mathrm{GELU} \to \mathrm{fc}_2 \to 1$ and the query coordinates concatenated to the input; modes and width are $16/64$ in 1D, $12/32$ in 2D and $8/20$ in 3D. Non-periodic axes are padded by 10% of the grid, symmetrically on all three axes in 3D. FNO is used only where a is a field over the domain: the sensor values are linearly interpolated onto the solve grid, FNO runs on that grid, and its output is interpolated linearly to the query points, the identity except on 1D Darcy, whose grid has 401 points against 201 queries.

Optimization. Adam, learning rate 10^{-3}, minibatch 20, step decay $\times 0.5$, weight decay 0 for DeepONet (except the coefficient network β_θ on Pipe, which takes 10^{-3} in PRIOR-STRUCT, PRIOR-RAND and PRIOR-POD alike) and 10^{-4} for FNO, and the per-sample relative L_2 loss of the PINO release. The step budget is fixed in advance from the epoch counts of the two reference implementations (2000 epochs for DeepONet, 500 for FNO) as epochs $\times \lceil \max(N, 200)/20 \rceil$: a cell with $N < 200$ receives the same number of steps as $N = 200$ (20,000 and 5,000) rather than fewer, and the budget grows with the data above it (50,000 and 100,000 DeepONet steps at $N = 500$ and 1000, 12,500 FNO steps at $N = 500$). There is no validation set and no checkpoint selection: training runs the fixed budget and the final model is evaluated, so N is the entire label budget. Identical for every formulation.

Seeds and splits. Five seeds $\{0, 7, 17, 23, 42\}$ per cell; the seed fixes both the draw of the N training instances and the network initialization. The held-out set is drawn once per benchmark and shared by every formulation, budget and seed (sizes in Table 9). The reported error is the mean over held-out instances of the per-instance relative L_2 error $\|\hat{u}_i - u_i\|_2 / \|u_i\|_2$; every cell also records the norm ratio $\|\hat{u}\|/\|u\|$ used in Table 12.

Residual-loss formulations. The PDE residual uses the *same discretizations as the prior solver*, is normalized per sample by $|f|$, and adds boundary terms separately. The forcing seen by the residual is the same sensor interpolation the prior sees, not the true grid forcing. The weight λ is swept over $\{10^{-10}, 10^{-8}, 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1\}$ and selected per seed by lowest held-out error, i.e. an oracle (Appendix C.1). The smallest value, 10^{-10}, approximates the $\lambda \to 0$ limit.

Initialization of the corrector. Under DeepONet the last linear layer of the corrector is initialized to zero, so the prediction at step 0 is u_p plus the scalar output offset, and the same holds for the two scalar gates on the branches of PRIOR-STRUCT (the $+2$ in Table 17). Under FNO the corrector keeps its standard initialization and its output is scaled by $\mathrm{std}(u)$ on the training set. Zero-initializing the FNO head instead starts the spectral body with no data gradient, and while the head grows the small

gradient that reaches the body loses to weight decay, so the body decays to zero: on Pipe this pinned the prediction at u_p in 23 of 25 runs, and scaling the output by the standard deviation of the residual rather than of u does the same on any benchmark where the prior resolves most of the field.

Structural injection. PRIOR-STRUCT adds to the PRIOR model a second branch whose trunk is not learned but given by $\Phi(x; a)$, the basis the solution of $\mathcal{F}_0$ exposes. The PRIOR model is retained unchanged inside it as the generic corrector ϵ_θ of Eq. 12, so deleting the Φ branch recovers PRIOR exactly and PRIOR-STRUCT carries more parameters than PRIOR (Table 10). The two branches are trained with the staged loss of Eq. 13: a joint loss on their sum lets them grow and cancel, which on optical drove both to 1.9 and 2.2 times the norm of the residual they were fitting. PRIOR-RAND and PRIOR-POD keep u_p and the source factor F of Eq. 14 and replace the spatial factor S, by a fixed random basis of the same width and by the leading POD modes of the training residuals zero-padded to that width, both rescaled to the norm of $\Phi_{\mathcal{F}_0}$ on the training set, and both trained with the same staged loss. Architecture and parameter count stay identical to PRIOR-STRUCT, so these two, not PRIOR, are the capacity-matched comparisons. Appendix F measures what ϵ_θ contributes.

Hardware. Three NVIDIA RTX 3090 GPUs (24 GB each), an Intel Core i9-10980XE and 125 GB RAM; PyTorch 2.14.0 with CUDA 12.6. Runs are spread over the three GPUs; FNO, PI and PRIOR-STRUCT runs repeated on each GPU agree bit for bit.

Benchmarks with no residual loss. Seven of the eighteen benchmarks carry no PI column in Table 11, for reasons that differ. On the two *Burgers* benchmarks the queried output is a single time slice of u, so u_t cannot be formed from the prediction and the evolution residual $u_t + uu_x - \nu u_{xx}$ cannot be assembled; the other 1D benchmarks are steady or queried on the full space–time grid, which is why they do admit one. On the two *Pipe* benchmarks the target is the streamwise velocity alone, so the momentum residual, which couples it to the pressure and the transverse velocity, cannot be evaluated on what the predictor emits. On *elasticity* the equilibrium residual is assemblable but non-identifying. Our predictor emits the three stress components independently, so the penalty is not vacuous on its search space (a field with $\sigma_{xx} = x$ alone has $\nabla\cdot\sigma \neq 0$), and it pushes predictions toward the equilibrium subspace. But that subspace holds the truth, the prior and the zero field alike, and what selects among them is boundary data the penalty does not carry. On *optical* the reference is generated from tabulated dispersive $n_j(\lambda)$ and $k_j(\lambda)$, so the discrepancy the learner must close is measured material data; the query variable is wavelength and no differential operator in λ relates the reflectance values at different wavelengths, so there is nothing to penalize. AIRFRANS is treated in Section 4.3. Elasticity and optical nonetheless expose a representation, and that representation is still usable, which is the point of separating the loss level from the prior and representation levels.

Table 11: **Raw results for the formulations without Φ.** Relative L_2 error, mean over five seeds, sample standard deviation in parentheses. λ^* is the oracle residual weight, selected per seed on the evaluation set; the column gives the value chosen most often, the smallest of them when several tie. PI is run at $N \leq 200$, to N=1000 on 2D Poisson, 3D layered and annulus, and at N=200 only on Darcy (Li et al., 2021); "–" marks a formulation not run in that cell.

Benchmark	Op.	N	RAW	PI	λ^*	PRIOR
1D nonlinear reaction	DeepONet	10	0.3934 (0.1024)	0.3646 (0.1119)	10^{-2}	0.0040 (0.0017)
		25	0.1261 (0.0236)	0.1218 (0.0239)	10^{-8}	0.0034 (0.0010)
		50	0.0571 (0.0080)	0.0560 (0.0073)	10^{-4}	0.0018 (0.0002)
		100	0.0386 (0.0078)	0.0358 (0.0071)	10^{-4}	0.0011 (0.0002)
		200	0.0246 (0.0029)	0.0230 (0.0024)	10^{-6}	0.0004 (0.0001)
		500	0.0134 (0.0020)	–	–	0.0001 (0.0000)
	FNO	10	0.3914 (0.0909)	0.3188 (0.0826)	10^{0}	0.0030 (0.0006)
		25	0.1074 (0.0214)	0.0638 (0.0103)	10^{0}	0.0019 (0.0002)
		50	0.0335 (0.0082)	0.0204 (0.0085)	10^{0}	0.0011 (0.0004)
		100	0.0099 (0.0014)	0.0078 (0.0011)	10^{-1}	0.0009 (0.0004)
		200	0.0064 (0.0007)	0.0051 (0.0004)	10^{-2}	0.0006 (0.0001)
		500	0.0045 (0.0007)	–	–	0.0004 (0.0001)
1D Burgers	DeepONet	10	0.4941 (0.0938)	–	–	0.2244 (0.0285)
		25	0.2916 (0.0273)	–	–	0.1762 (0.0076)
		50	0.1930 (0.0122)	–	–	0.1376 (0.0066)

Benchmark	Op.	N	RAW	PI	λ^*	PRIOR
		100	0.1479 (0.0063)	–	–	0.0994 (0.0037)
		200	0.1174 (0.0049)	–	–	0.0704 (0.0023)
		500	0.0765 (0.0021)	–	–	0.0416 (0.0015)
	FNO	10	0.2010 (0.0436)	–	–	0.1342 (0.0204)
		25	0.0665 (0.0038)	–	–	0.0463 (0.0052)
		50	0.0267 (0.0039)	–	–	0.0207 (0.0020)
		100	0.0103 (0.0013)	–	–	0.0086 (0.0009)
		200	0.0064 (0.0004)	–	–	0.0047 (0.0003)
		500	0.0031 (0.0001)	–	–	0.0025 (0.0001)
1D heat	DeepONet	10	0.4015 (0.1314)	0.3793 (0.1191)	10^{-8}	0.1989 (0.0817)
		25	0.1514 (0.0297)	0.1447 (0.0314)	10^{-10}	0.0548 (0.0190)
		50	0.0723 (0.0133)	0.0652 (0.0080)	10^{-10}	0.0264 (0.0074)
		100	0.0457 (0.0093)	0.0478 (0.0132)	10^{-8}	0.0163 (0.0056)
		200	0.0310 (0.0070)	0.0300 (0.0047)	10^{-8}	0.0139 (0.0044)
		500	0.0151 (0.0022)	–	–	0.0047 (0.0014)
	FNO	10	0.3364 (0.1044)	0.3199 (0.0837)	10^{-8}	0.2307 (0.0680)
		25	0.0996 (0.0090)	0.0912 (0.0075)	10^{-6}	0.0633 (0.0113)
		50	0.0339 (0.0063)	0.0303 (0.0084)	10^{-6}	0.0183 (0.0035)
		100	0.0139 (0.0019)	0.0127 (0.0013)	10^{-4}	0.0105 (0.0018)
		200	0.0109 (0.0011)	0.0099 (0.0009)	10^{-8}	0.0087 (0.0015)
		500	0.0069 (0.0011)	–	–	0.0039 (0.0003)
1D convection	DeepONet	10	0.3645 (0.0825)	0.3373 (0.0748)	10^{-8}	0.0739 (0.0238)
		25	0.1494 (0.0303)	0.1444 (0.0263)	10^{-8}	0.0307 (0.0128)
		50	0.0745 (0.0056)	0.0732 (0.0042)	10^{-3}	0.0161 (0.0023)
		100	0.0482 (0.0045)	0.0457 (0.0025)	10^{-8}	0.0098 (0.0009)
		200	0.0328 (0.0011)	0.0305 (0.0010)	10^{-4}	0.0081 (0.0005)
		500	0.0187 (0.0015)	–	–	0.0048 (0.0003)
	FNO	10	0.4428 (0.0940)	0.2759 (0.0635)	10^{0}	0.0770 (0.0150)
		25	0.1144 (0.0333)	0.1120 (0.0334)	10^{-3}	0.0194 (0.0029)
		50	0.0352 (0.0051)	0.0344 (0.0049)	10^{-10}	0.0067 (0.0020)
		100	0.0122 (0.0017)	0.0118 (0.0017)	10^{-5}	0.0022 (0.0004)
		200	0.0070 (0.0015)	0.0063 (0.0003)	10^{-6}	0.0017 (0.0001)
		500	0.0039 (0.0003)	–	–	0.0010 (0.0002)
battery	DeepONet	10	0.4721 (0.1248)	0.4529 (0.1197)	10^{-6}	0.2970 (0.0645)
		25	0.1821 (0.0343)	0.1729 (0.0320)	10^{-6}	0.1471 (0.0256)
		50	0.0928 (0.0146)	0.0869 (0.0123)	10^{-6}	0.0853 (0.0232)
		100	0.0531 (0.0039)	0.0506 (0.0028)	10^{-10}	0.0541 (0.0052)
		200	0.0321 (0.0014)	0.0294 (0.0035)	10^{-6}	0.0264 (0.0021)
		500	0.0126 (0.0018)	–	–	0.0101 (0.0006)
	FNO	10	0.4949 (0.1204)	0.4598 (0.1068)	10^{-3}	0.3185 (0.0424)
		25	0.2717 (0.0501)	0.2410 (0.0185)	10^{-3}	0.1731 (0.0144)
		50	0.1484 (0.0312)	0.1277 (0.0325)	10^{-3}	0.1132 (0.0223)
		100	0.0683 (0.0092)	0.0641 (0.0072)	10^{-10}	0.0568 (0.0108)
		200	0.0345 (0.0072)	0.0312 (0.0073)	10^{-4}	0.0297 (0.0053)
		500	0.0174 (0.0020)	–	–	0.0138 (0.0017)
1D Darcy	DeepONet	10	0.1694 (0.0363)	0.1510 (0.0169)	10^{-5}	0.1490 (0.0361)
		25	0.0769 (0.0064)	0.0744 (0.0046)	10^{-6}	0.0824 (0.0127)
		50	0.0550 (0.0080)	0.0538 (0.0047)	10^{-10}	0.0573 (0.0036)
		100	0.0372 (0.0028)	0.0362 (0.0025)	10^{-10}	0.0428 (0.0020)
		200	0.0271 (0.0014)	0.0266 (0.0024)	10^{-6}	0.0328 (0.0007)
		500	0.0190 (0.0016)	–	–	0.0241 (0.0009)
	FNO	10	0.1050 (0.0319)	0.0847 (0.0224)	10^{-2}	0.1446 (0.0324)
		25	0.0377 (0.0051)	0.0338 (0.0035)	10^{-2}	0.0581 (0.0104)
		50	0.0175 (0.0014)	0.0161 (0.0012)	10^{-3}	0.0285 (0.0039)
		100	0.0097 (0.0005)	0.0090 (0.0005)	10^{-4}	0.0129 (0.0020)

Benchmark	Op.	N	RAW	PI	λ^*	PRIOR
		200	0.0061 (0.0007)	0.0056 (0.0005)	10^{-4}	0.0077 (0.0010)
		500	0.0037 (0.0003)	–	–	0.0037 (0.0006)
2D Darcy	DeepONet	10	0.3518 (0.0303)	0.3369 (0.0113)	10^{-2}	0.1607 (0.0032)
		25	0.2507 (0.0128)	0.2440 (0.0107)	10^{-10}	0.1360 (0.0040)
		50	0.1967 (0.0043)	0.1890 (0.0071)	10^{-2}	0.1149 (0.0031)
		100	0.1487 (0.0054)	0.1423 (0.0029)	10^{-2}	0.0954 (0.0009)
		200	0.1116 (0.0028)	0.1094 (0.0018)	10^{-3}	0.0821 (0.0009)
		500	0.0859 (0.0012)	–	–	0.0734 (0.0019)
	FNO	10	0.1361 (0.0500)	0.1347 (0.0506)	10^{-4}	0.1962 (0.1729)
		25	0.0665 (0.0026)	0.0660 (0.0024)	10^{-8}	0.0803 (0.0089)
		50	0.0466 (0.0014)	0.0458 (0.0016)	10^{-2}	0.0548 (0.0019)
		100	0.0298 (0.0014)	0.0294 (0.0015)	10^{-3}	0.0338 (0.0021)
		200	0.0213 (0.0014)	0.0210 (0.0013)	10^{-3}	0.0217 (0.0009)
		500	0.0136 (0.0004)	–	–	0.0128 (0.0009)
3D Darcy	DeepONet	10	0.5291 (0.0209)	0.5130 (0.0151)	10^{-2}	0.1792 (0.0026)
		25	0.4533 (0.0225)	0.4376 (0.0187)	10^{-8}	0.1712 (0.0026)
		50	0.3798 (0.0072)	0.3654 (0.0088)	10^{-1}	0.1543 (0.0019)
		100	0.2852 (0.0091)	0.2773 (0.0060)	10^{-2}	0.1366 (0.0023)
		200	0.2027 (0.0016)	0.1971 (0.0020)	10^{-2}	0.1235 (0.0010)
		500	0.1491 (0.0046)	–	–	0.1105 (0.0013)
	FNO	10	0.5789 (0.4079)	0.2989 (0.2550)	10^{-1}	0.6839 (0.3769)
		25	0.0531 (0.0038)	0.0528 (0.0039)	10^{-2}	0.0601 (0.0056)
		50	0.0384 (0.0020)	0.0383 (0.0021)	10^{-3}	0.0426 (0.0022)
		100	0.0282 (0.0004)	0.0280 (0.0004)	10^{-3}	0.0290 (0.0004)
		200	0.0238 (0.0004)	0.0238 (0.0004)	10^{-10}	0.0235 (0.0006)
		500	0.0187 (0.0005)	–	–	0.0183 (0.0003)
2D Poisson	DeepONet	10	0.7007 (0.0363)	0.6824 (0.0403)	10^{-2}	0.1169 (0.0175)
		25	0.5369 (0.0205)	0.5230 (0.0227)	10^{-3}	0.0704 (0.0092)
		50	0.3995 (0.0142)	0.3811 (0.0132)	10^{-2}	0.0408 (0.0031)
		100	0.2944 (0.0114)	0.2798 (0.0112)	10^{-1}	0.0231 (0.0007)
		200	0.2240 (0.0048)	0.2127 (0.0053)	10^{-1}	0.0161 (0.0006)
		500	0.1109 (0.0074)	0.1054 (0.0076)	10^{-5}	0.0104 (0.0004)
		1000	0.0490 (0.0011)	0.0481 (0.0016)	10^{-4}	0.0061 (0.0003)
3D layered	DeepONet	10	1.1386 (0.1090)	1.0448 (0.0565)	10^{0}	0.1908 (0.0069)
		25	1.2340 (0.0319)	1.0255 (0.0258)	10^{0}	0.1995 (0.0097)
		50	1.3825 (0.1372)	1.0343 (0.0257)	10^{0}	0.2147 (0.0121)
		100	1.5477 (0.1078)	1.0379 (0.0170)	10^{0}	0.2295 (0.0096)
		200	1.6102 (0.0816)	1.0350 (0.0295)	10^{0}	0.2377 (0.0096)
		500	1.6093 (0.0592)	1.0346 (0.0120)	10^{0}	0.2516 (0.0050)
		1000	1.6392 (0.0880)	1.0306 (0.0188)	10^{0}	0.2641 (0.0041)
annulus	DeepONet	10	0.9963 (0.0426)	0.9139 (0.0438)	10^{-4}	0.6866 (0.0250)
		25	0.8292 (0.0432)	0.7906 (0.0344)	10^{-6}	0.6181 (0.0077)
		50	0.7829 (0.0171)	0.7177 (0.0198)	10^{-4}	0.6386 (0.0136)
		100	0.6901 (0.0285)	0.6379 (0.0105)	10^{-5}	0.6082 (0.0139)
		200	0.5774 (0.0086)	0.5650 (0.0158)	10^{-5}	0.5195 (0.0204)
		500	0.4373 (0.0053)	0.4285 (0.0028)	10^{-8}	0.4491 (0.0084)
		1000	0.3941 (0.0109)	0.3789 (0.0182)	10^{-8}	0.3962 (0.0087)
elasticity	DeepONet	10	0.9101 (0.0322)	–	–	0.2749 (0.0157)
		25	0.7523 (0.0256)	–	–	0.2058 (0.0234)
		50	0.6340 (0.0132)	–	–	0.1534 (0.0075)
		100	0.5699 (0.0076)	–	–	0.1306 (0.0019)
		200	0.5223 (0.0101)	–	–	0.1098 (0.0018)
		500	0.4172 (0.0034)	–	–	0.0765 (0.0009)
		1000	0.3426 (0.0031)	–	–	0.0571 (0.0016)
optical	DeepONet	10	0.5885 (0.0774)	–	–	0.3796 (0.0830)

Benchmark	Op.	N	RAW	PI	λ^*	PRIOR
		25	0.3455 (0.0331)	–	–	0.2402 (0.0293)
		50	0.2831 (0.0155)	–	–	0.2152 (0.0155)
		100	0.2118 (0.0073)	–	–	0.1652 (0.0088)
		200	0.1605 (0.0173)	–	–	0.1229 (0.0087)
		500	0.0920 (0.0095)	–	–	0.0600 (0.0034)
		1000	0.0573 (0.0048)	–	–	0.0367 (0.0020)
Darcy (Li et al., 2021)	DeepONet	10	0.2794 (0.0228)	–	–	0.1643 (0.0172)
		25	0.2023 (0.0110)	–	–	0.1289 (0.0052)
		50	0.1738 (0.0048)	–	–	0.1064 (0.0030)
		100	0.1291 (0.0026)	–	–	0.0878 (0.0009)
		200	0.0989 (0.0040)	0.0950 (0.0020)	10^{-2}	0.0716 (0.0007)
		500	0.0751 (0.0018)	–	–	0.0625 (0.0009)
	FNO	10	0.1060 (0.0098)	–	–	0.1253 (0.0154)
		25	0.0650 (0.0030)	–	–	0.0769 (0.0042)
		50	0.0429 (0.0024)	–	–	0.0536 (0.0016)
		100	0.0268 (0.0016)	–	–	0.0313 (0.0013)
		200	0.0185 (0.0010)	0.0184 (0.0010)	10^{-3}	0.0193 (0.0011)
		500	0.0121 (0.0003)	–	–	0.0119 (0.0005)
Burgers (Li et al., 2021)	DeepONet	10	0.5357 (0.0596)	–	–	0.3613 (0.0204)
		25	0.3665 (0.0257)	–	–	0.2793 (0.0193)
		50	0.2827 (0.0289)	–	–	0.2291 (0.0054)
		100	0.2281 (0.0135)	–	–	0.1974 (0.0072)
		200	0.1934 (0.0084)	–	–	0.1676 (0.0041)
		500	0.1579 (0.0052)	–	–	0.1243 (0.0040)
	FNO	10	0.1213 (0.0181)	–	–	0.1283 (0.0182)
		25	0.0308 (0.0059)	–	–	0.0355 (0.0088)
		50	0.0137 (0.0016)	–	–	0.0149 (0.0020)
		100	0.0056 (0.0005)	–	–	0.0061 (0.0007)
		200	0.0038 (0.0003)	–	–	0.0035 (0.0002)
		500	0.0023 (0.0005)	–	–	0.0020 (0.0001)
Pipe (Li et al., 2023)	DeepONet	10	0.2464 (0.0142)	–	–	0.1039 (0.0057)
		25	0.1788 (0.0022)	–	–	0.0884 (0.0022)
		50	0.1460 (0.0061)	–	–	0.0738 (0.0017)
		100	0.1203 (0.0022)	–	–	0.0598 (0.0037)
		200	0.1009 (0.0058)	–	–	0.0434 (0.0025)
		500	0.0413 (0.0027)	–	–	0.0325 (0.0013)
		1000	0.0310 (0.0017)	–	–	0.0261 (0.0006)
	FNO	10	0.1198 (0.0046)	–	–	0.0844 (0.0051)
		25	0.0778 (0.0061)	–	–	0.0582 (0.0038)
		50	0.0525 (0.0040)	–	–	0.0455 (0.0031)
		100	0.0273 (0.0028)	–	–	0.0312 (0.0054)
		200	0.0169 (0.0015)	–	–	0.0196 (0.0019)
Pipe-GRF	DeepONet	10	0.3194 (0.0218)	–	–	0.1093 (0.0046)
		25	0.2183 (0.0064)	–	–	0.0901 (0.0028)
		50	0.2049 (0.0079)	–	–	0.0885 (0.0010)
		100	0.1838 (0.0069)	–	–	0.0832 (0.0027)
		200	0.1606 (0.0079)	–	–	0.0791 (0.0024)
		500	0.0960 (0.0048)	–	–	0.0646 (0.0039)
	FNO	10	0.1232 (0.0127)	–	–	0.0789 (0.0112)
		25	0.0723 (0.0058)	–	–	0.0477 (0.0039)
		50	0.0627 (0.0038)	–	–	0.0439 (0.0027)
		100	0.0561 (0.0045)	–	–	0.0400 (0.0020)
		200	0.0484 (0.0029)	–	–	0.0385 (0.0015)
		500	0.0338 (0.0032)	–	–	0.0309 (0.0022)

Table 12: **When PI and RAW return a field near zero.** Error and norm ratio $\|\hat{u}\|/\|u\|$ at each weight on the grid, averaged over five seeds at $N = 200$, for the two benchmarks on which a formulation scores close to the error of a zero prediction (1.00). The last block is RAW on 3D layered across budgets, whose error exceeds 1: a full-norm prediction uncorrelated with the target.

Benchmark	λ	rel. L_2	$\|\hat{u}\|/\|u\|$
3D layered	10^{-10}	1.6022	1.1314
	10^{-8}	1.5794	1.1193
	10^{-6}	1.5985	1.1293
	10^{-5}	1.5623	1.0799
	10^{-4}	1.5095	1.0226
	10^{-3}	1.3544	0.7792
	10^{-2}	1.2873	0.6575
	10^{-1}	1.1430	0.4319
	10^{0}	1.0350	0.2129
annulus	10^{-10}	0.5753	1.0035
	10^{-8}	0.5786	0.9942
	10^{-6}	0.5806	0.9846
	10^{-5}	0.5664	0.9490
	10^{-4}	0.6189	0.7995
	10^{-3}	0.8635	0.3620
	10^{-2}	0.9803	0.0560
	10^{-1}	0.9994	0.0021
	10^{0}	1.0000	0.0005
3D layered (RAW)	$N = 10$	1.1386	0.3951
	$N = 25$	1.2340	0.5652
	$N = 50$	1.3825	0.7974
	$N = 100$	1.5477	1.0372
	$N = 200$	1.6102	1.1532
	$N = 500$	1.6093	1.2093
	$N = 1000$	1.6392	1.2809

C.3 WHEN PI AND RAW APPROACH THE ZERO FIELD

On 3D layered and annulus, two formulations sit near or above the error a prediction of zero would produce. RAW and PI carry no prior, so a relative error of exactly 1.00 is what the zero field gives, and a number near 1.00 can mean either a fit that has learned nothing or an abstention. To separate the two we record, alongside the error, the norm ratio $\|\hat{u}\|/\|u\|$ of the prediction on the held-out set; Table 12 reports it across the λ grid at $N = 200$.

The two failures are opposite. PI abstains: as λ grows its norm ratio falls to 0.21 on 3D layered and to 0.0005 on annulus, and the error converges to 1.00 (from above on 3D layered, from below on annulus): the residual of $\mathcal{F}_0$ is minimized by the zero field, and on 3D layered, where the oracle selects λ=1, the label term cannot hold the prediction away from it; on annulus the oracle stays clear of this regime ($\lambda^* \leq 10^{-4}$ at every budget). RAW does the reverse: on 3D layered its norm ratio rises from 0.40 at N=10 to 1.28 at N=1000 while its error rises from 1.14 to 1.64, a full-scale prediction uncorrelated with the target. The error rises with N because the norm does while the correlation stays near zero: an uncorrelated prediction at norm ratio ρ has error about $\sqrt{1+\rho^2}$, which gives 1.08 at N=10 and 1.63 at N=1000 against the measured 1.14 and 1.64. The conductivities and interface resistance are fixed across instances; the solution depends linearly on the source and strongly nonlinearly on the layer thicknesses, and even 1000 labels are not enough for the backbone to find that structure; the equation supplies it for free, and PRIOR-STRUCT reaches 0.0347 at N=200.

C.4 PRIOR AGAINST LOSS, CELL BY CELL

A cell is a (benchmark, backbone, N) triple of the thirteen pairs of Table 13 in which both arms ran on the same five seeds, with PI at its per-seed oracle λ. Paired over those seeds, PRIOR is more accurate at $p < 0.05$ in 40 of the 63 cells and PI in 15; the remaining eight are unresolved. Every cell PI wins but one is in the Darcy family (1D Darcy under both backbones, 2D and 3D Darcy under FNO), where the construction leaves $\varepsilon_{\text{prior}} \approx 0.19$–$0.22$; the exception is annulus under DeepONet at N=500 ($p = 0.004$). With five seeds these tests have limited power, so nonsignificant cells should not be interpreted as evidence of no difference.

Table 13: **The prior-versus-loss margin, by how much the prior resolves.** The thirteen benchmark–architecture pairs that admit both formulations, excluding the three linear benchmarks, sorted by $\varepsilon_{\text{prior}}$ and split at its median; each pair contributes one cell per budget it was run at (63 cells, five seeds each). 1D heat, 1D convection and battery, whose map is exactly linear, are excluded; the last row adds them back. PI is at its oracle weight, so the margin is a lower bound on what a practical λ rule would give.

Stratum	$\varepsilon_{\text{prior}}$	pairs	cells	PRIOR better	geo. mean PI/PRIOR
resolves more	0.003–0.190	6	34	30/34	6.48×
resolves less	0.196–0.665	7	29	12/29	0.99×
all	0.003–0.665	13	63	42/63	2.73×
all + linear three	0.003–0.665	19	93	71/93	2.52×

The upper stratum of Table 13, the seven pairs with $\varepsilon_{\text{prior}} \geq 0.196$, is 1D Darcy, 2D Darcy and Darcy (Li et al., 2021) under both backbones together with annulus; it holds 17 of the 21 cells the prior loses, including annulus at N=500 and 1000. The other four are 3D Darcy under FNO at $N \leq 100$, where the three formulations are within 0.008 of each other from N=25 on and individual seeds diverge at N=10 (standard deviations above 0.25; Table 11).

C.5 EXPOSED BASIS ORDER BEHAVES LIKE CAPACITY

The basis is not free: its width is a number of coefficients the budget has to estimate. Of the six benchmarks that expose Φ, PRIOR-STRUCT is clearly above the PRIOR model it extends on exactly one, *elasticity*, and only at the smallest budgets (on optical the two do not separate): 0.4317 against 0.2749 at N=10, 0.2978 against 0.2058 at N=25 and 0.1838 against 0.1534 at N=50; it crosses at N=100 (0.1008 against 0.1306) and stays ahead (0.0349 against 0.0571 at N=1000).

The cause is the coefficient map, not the corrector. Elasticity is the one benchmark whose basis does not separate by mode: the non-circular hole mixes the Laurent orders, so each of the 8 load coefficients couples to all 64 stress fields and β_θ must learn the full 64×8 coupling $W(a_g)$ from three shape coefficients, where elsewhere each input coefficient multiplies only its own mode. Deleting the generic corrector changes nothing (PRIOR-STRUCT$_{\text{no-free}}$ is 0.4311 against 0.4317 at N=10, Table 17), while narrowing the basis does: the construction used here truncates the Laurent orders to $|k| \leq 8$ and the load modes to $j \leq 2$, giving $n_b = 512$ instead of 4096 ($|k| \leq 16$, $j \leq 8$), without which the branch fits the training set (0.012) and not the test set (0.183) at N=200. Exposed width therefore behaves like capacity: it buys expressiveness and charges samples for it. We report one truncation for all budgets; selecting it per budget would tune the method to the budget. Appendix C.6 widens the coefficient map instead.

C.6 RAISING THE DIMENSION OF THE COEFFICIENT MAP

On the structural benchmarks β_θ reads a geometry vector of at most nine numbers, so the structured branch could be read as a low-dimensional regression dressed up as operator learning. The annulus admits a clean test of that reading: its geometry is the set of $L-1$ interface radii, so raising the number of regions raises the dimension of β_θ's domain and leaves everything else in place. We build $L \in \{5, 9, 17\}$, giving domains of 4, 8 and 16 dimensions and bases of n_b = 216, 408 and 792 columns, with the conductivities tiled periodically so that the material contrast, the source of the effect being learned, does not change with L. The default benchmark is L=5, and the other two are additional registrations, so the L=5 results are bit-identical to those in the main text.

Table 14: **Annulus with a wider coefficient map (N=200, DeepONet).** Relative L_2 error, mean over five seeds. $\dim a_g$ is the number of interface radii β_θ reads. $\varepsilon_{\text{prior}}$ is the error of u_p alone, and an error of one is what a prediction of zero gives.

	$\dim a_g$	n_b	$\varepsilon_{\text{prior}}$	RAW	PRIOR	PRIOR-STRUCT	PRIOR-RAND	PRIOR-POD
annulus (L=5)	4	216	0.6649	0.5774	0.5195	**0.0251**	0.5568	0.5738
annulus (L=9)	8	408	0.4651	0.4912	0.4316	**0.0469**	0.4587	0.4808
annulus (L=17)	16	792	2.9430	1.0505	1.7106	**0.4631**	2.0394	1.9114

The ordering does not depend on the dimension (Table 14). PRIOR-STRUCT is the most accurate arm at all three, and its margin over the best of the other four arms narrows from $20.7\times$ at $\dim a_g{=}4$ to $9.2\times$ at 8 and $2.3\times$ at 16 without reversing. The problem itself gets harder as L grows: at $L{=}17$ the prior alone is at 2.94 and every formulation except PRIOR-STRUCT is at or above 1.00, the error a prediction of zero would give, while PRIOR-STRUCT is at 0.46. What the wider domain costs is visible in absolute terms (0.0251 to 0.4631), but the same widening moves the controls further, not closer.

Two qualifications. At $L{=}17$ PRIOR-STRUCT does not improve with data in the range we ran (0.463, 0.439, 0.482 and 0.463 at $N{=}25$, 50, 100 and 200), so the sixteen-dimensional map is not being estimated better with more labels at these budgets; the gain there comes from the basis, not from learning the coefficients well. And deleting the generic corrector helps slightly rather than hurting (0.4135 at $L{=}17$ and 0.0432 at $L{=}9$, against 0.4631 and 0.0469), as Appendix F reports for annulus at $L{=}5$.

D THE PRIOR'S VALUE BEYOND PREDICTION

The value of the prior is not exhausted by its accuracy as a predictor. On several benchmarks of Table 9, u_p before any residual is learned is worse than RAW at the smallest budget, and only the composition beats both: at $N{=}10$ (Table 11), for example, the battery benchmark's prior error is 0.5220 against a DEEPONET RAW of 0.4721, and the one-dimensional heat benchmark's prior error is 0.4930 against a RAW of 0.4015, with the composed predictor PRIOR reaching 0.2970 and 0.1989 respectively. The 3D layered benchmark makes the point in reverse: there u_p alone is 0.1751 while RAW is above 1.1 at every budget, so the prior is the more accurate predictor by a wide margin and yet the PRIOR formulation built on it is worse than u_p itself (0.2377 at $N{=}200$); only the structured branch converts the same construction into an improvement.

E AIRFRANS IN FULL

This appendix gives the AIRFRANS data, what the prior leaves, and how the prior compares with data-derived peers. Every number uses the split, seeds and per-sample metric of Table 4.

Dataset and grid. We use all 1000 cases of the public dataset. The pressure, divided by density, is sampled on a 60×40 Cartesian grid around the airfoil; the 181 grid points inside any of the 1000 airfoils are dropped from every case, leaving the $n_x = 2219$ query points of Section 4.3. A fixed seeded permutation holds out 200 cases for evaluation and leaves 800 as the training pool, from which each seed draws its $N \leq 500$ training cases. The input a stacks U_∞, α and the upper- and lower-surface ordinates at 16 chordwise stations, 34 numbers standardised by their mean and standard deviation over the 1000 cases, which involves no labels. Over the dataset $U_\infty \in [31.28,\, 93.59]\,\mathrm{m\,s^{-1}}$ and $\alpha \in [-4.94^\circ,\, 14.93^\circ]$.

Why the scaling law is approximate. The factorisation of Eq. 17 is exact; Eq. 18 drops the Reynolds argument of C_p, and AIRFRANS does not hold Re fixed: the kinematic viscosity is constant while U_∞ varies by a factor of 2.99, so at unit chord Re runs from 2.0×10^6 to 6.0×10^6. Under $v \mapsto \kappa v$ at fixed ν the inertial and pressure terms scale as κ^2 and $\nu\nabla^2 v$ as κ, while the eddy viscosity, proportional to the velocity scale, scales as κ and leaves $\nabla\cdot(\nu_t \nabla v)$ at κ^2. The defect is therefore confined to where molecular viscosity is not negligible against ν_t, the near-wall region at these Reynolds numbers. It is left to the residual corrector, as is the linearisation in α.

What the prior leaves. The residual is $r = q[C_p - c_0 - c_1\alpha]$, so the prior removes the leading-order flow-condition dependence but not the q scaling of the residual's magnitude. With u_p fitted on the 800-case training pool and evaluated on the 200 held-out cases, a least-squares regression on $(1, q, q\alpha)$ explains $R^2 = 0.020$ of the variance of r, against $R^2 = 0.896$ for $p^\star$; adding $q\alpha^2$, $q \log U_\infty$ and U_∞ raises the former only to $R^2 = 0.106$. The magnitude still scales with dynamic pressure, $\mathrm{corr}(\|r_i\|, q_i) = 0.72$, and dividing it out removes the speed dependence, $\mathrm{corr}(\|r_i\|/q_i,\, U_{\infty,i}) = -0.03$, which is what CP-PRIOR does by construction. Over the held-out cases $\|r_i\|/\|p_i^\star\|$ has median 0.260 and mean 0.354, the value $\varepsilon_{\mathrm{prior}}$ reports; the distribution has a long right tail, and a few cases with large error pull the mean up.

Table 15: **The AIRFRANS prior against data-derived peers.** Per-sample relative L_2 error over the 200 held-out cases, mean over the five seeds of Table 4; bold marks the row minimum. *Data only* is the most accurate of the three peers of Appendix G on each seed, chosen on the held-out error; *ratio* is Data only over PRIOR, and *wins* counts the seeds in which PRIOR is more accurate.

Budget	Prior alone	Data only	PRIOR	Ratio	Wins
$N = 10$	0.4166	1.0490	**0.3958**	2.65	5/5
$N = 25$	0.3788	0.6948	**0.2005**	3.46	5/5
$N = 50$	0.3712	0.4292	**0.1269**	3.38	5/5
$N = 100$	0.3529	0.2134	**0.0801**	2.66	5/5
$N = 200$	0.3549	0.1499	**0.0507**	2.96	5/5
$N = 500$	0.3545	0.0860	**0.0299**	2.87	5/5

Is it the scaling law or the fit? The prior combines the $q(a)$ factor the equation supplies with a rank-two profile fitted from labels. CP-NET, $\hat{p} = q(a)\, r_\theta(x; a)$, keeps the factor and drops the profile; CP-PRIOR, $\hat{p} = u_p + q(a)\, r_\theta(x; a)$, keeps both. Both use the network, inputs, loss and budget of RAW, and CP-PRIOR starts at u_p as PRIOR does. At N=200 CP-NET reaches 0.0484 against 0.0854 for RAW and 0.0507 for PRIOR, and CP-PRIOR reaches 0.0292 (Table 4): the factor alone is worth as much as the fitted profile alone, and the two together are worth more than either.

A prior that consumes labels. AIRFRANS is the one benchmark whose prior is fitted, and since it also carries the paper's headline case for equation information without a residual, it is worth saying precisely what the equation contributes. The construction is the *indirect* branch of Eq. 9, which is part of the taxonomy rather than an exception to it: the equation fixes the functional form and the instance-side factor $\psi_{\mathcal{F}_0}(a) = q(a)(1, \alpha)^\top$ in closed form, before any label is seen, and the labels only fill in the spatial profile $c(x)$, two degrees of freedom per query point, by ordinary least squares. Those labels are the same N instances every formulation receives, so none is given data the others lack, and the fit is determined for $N \geq 2$, which is why the prior is available at the smallest budget in the sweep. Whether the equation or the fit carries the gain is testable, and the first two columns of Table 15 are the test. The data-derived peers receive strictly more than the prior does: the same N labelled instances, up to twenty basis degrees of freedom against our two, and the free-stream velocity and thirty-two surface coordinates rather than the dynamic pressure and the angle alone. They still lose to the prior alone by $2.52\times$ at N=10 and $1.83\times$ at N=25. A fitted rank-two model is not in itself a good prior here; it is a good prior because the rank-two structure and the $q(a)$ scaling are the ones the momentum equation prescribes. We nonetheless keep this benchmark apart from the seventeen label-free priors and flag it wherever it appears, and the counts of Sections 4.1 and 4.2 rest on the label-free ones alone.

The prior alone is capped; the composition is not. As a stand-alone predictor the prior beats the best peer in every seed up to N=25, in four of five at N=50 and in none from N=100: it is rank two by construction, so its error stops falling at N=100 while the peers keep improving. With the DeepONet corrector on top, PRIOR beats the best peer in every seed at every budget, by $2.65\times$ at N=10 and $2.87\times$ at N=500, with no trend in N. The peers are not neural networks, so this ratio changes the learner as well as the anchor; the comparison that holds the learner fixed is RAW against PRIOR in Table 4, where the margin falls from $3.50\times$ to $1.55\times$.

F WHAT THE GENERIC CORRECTOR CONTRIBUTES

Eq. 12 carries a generic corrector ϵ_θ alongside the structured branch, to absorb the part of the residual outside the span of $\Phi_{\mathcal{F}_0}$. How much it actually absorbs is measurable: we delete it and refit, holding the equation-derived branch, the data, the seeds, the budget and the optimiser fixed, so the only change is whether the model may represent anything outside $\operatorname{span} \Phi_{\mathcal{F}_0}$. The ablated model is exactly the structural injection written without its last term, $\hat{u}_\theta = u_p + \Phi_{\mathcal{F}_0} \beta_\theta$. Every cell of the PRIOR-STRUCT column was re-run in the same pass and agrees with Table 16 to within 5×10^{-3} relative, which is what makes the two columns of Table 17 comparable.

The corrector is inert wherever the basis is adequate. On 3D layered, elasticity and Pipe deleting it changes the error by at most 4% at every budget, and on annulus it is neutral or helps (by up to 11%), although it holds a large share of the parameters (73,350 of 235,374 on 3D layered, 28,500 of 59,914 on elasticity, 847,550 of 859,680 on Pipe). On 2D Poisson deleting it *improves* the error

Table 16: **Raw results for the formulations with Φ (DeepONet).** Relative L_2 error, mean over five seeds, sample standard deviation in parentheses. PRIOR is repeated from Table 11 so that each row reads as what the basis adds to the model it extends. Optical is the benchmark on which PRIOR-STRUCT, PRIOR and PRIOR-POD do not separate, and Pipe the one on which PRIOR-STRUCT and PRIOR-POD do not separate at N=200.

Benchmark	N	PRIOR	PRIOR-STRUCT	PRIOR-RAND	PRIOR-POD
2D Poisson	10	0.1169 (0.0175)	0.0162 (0.0138)	0.1180 (0.0174)	0.0687 (0.0135)
	25	0.0704 (0.0092)	0.0031 (0.0002)	0.0721 (0.0082)	0.0437 (0.0054)
	50	0.0408 (0.0031)	0.0027 (0.0001)	0.0423 (0.0021)	0.0257 (0.0035)
	100	0.0231 (0.0007)	0.0026 (0.0001)	0.0249 (0.0009)	0.0161 (0.0024)
	200	0.0161 (0.0006)	0.0025 (0.0001)	0.0179 (0.0007)	0.0134 (0.0004)
	500	0.0104 (0.0004)	0.0024 (0.0001)	0.0129 (0.0005)	0.0089 (0.0012)
	1000	0.0061 (0.0003)	0.0023 (0.0003)	0.0094 (0.0009)	0.0088 (0.0016)
3D layered	10	0.1908 (0.0069)	0.1363 (0.0244)	2.5940 (1.1869)	0.4882 (0.1389)
	25	0.1995 (0.0097)	0.0716 (0.0048)	0.7756 (0.1614)	0.4110 (0.0441)
	50	0.2147 (0.0121)	0.0543 (0.0025)	0.4462 (0.0629)	0.3628 (0.0817)
	100	0.2295 (0.0096)	0.0426 (0.0024)	0.2993 (0.0121)	0.2841 (0.0304)
	200	0.2377 (0.0096)	0.0347 (0.0023)	0.2633 (0.0090)	0.2253 (0.0179)
	500	0.2516 (0.0050)	0.0319 (0.0016)	0.2653 (0.0098)	0.2229 (0.0103)
	1000	0.2641 (0.0041)	0.0302 (0.0006)	0.2657 (0.0118)	0.2351 (0.0071)
annulus	10	0.6866 (0.0250)	0.0564 (0.0039)	3.2979 (2.3448)	1.5854 (0.4174)
	25	0.6181 (0.0077)	0.0357 (0.0025)	1.2189 (0.1889)	1.0616 (0.1748)
	50	0.6386 (0.0136)	0.0310 (0.0040)	0.8204 (0.0386)	0.9524 (0.1603)
	100	0.6082 (0.0139)	0.0265 (0.0016)	0.6870 (0.0088)	0.7643 (0.1674)
	200	0.5195 (0.0204)	0.0251 (0.0013)	0.5568 (0.0325)	0.5738 (0.0113)
	500	0.4491 (0.0084)	0.0214 (0.0007)	0.4889 (0.0551)	0.5192 (0.0439)
	1000	0.3962 (0.0087)	0.0201 (0.0003)	0.4609 (0.0829)	0.5031 (0.0764)
elasticity	10	0.2749 (0.0157)	0.4317 (0.0504)	0.7863 (0.4395)	1.2192 (0.1853)
	25	0.2058 (0.0234)	0.2978 (0.0202)	0.5822 (0.1841)	0.8485 (0.1105)
	50	0.1534 (0.0075)	0.1838 (0.0207)	0.3184 (0.0102)	0.6595 (0.0653)
	100	0.1306 (0.0019)	0.1008 (0.0083)	0.2062 (0.0215)	0.4106 (0.0345)
	200	0.1098 (0.0018)	0.0626 (0.0016)	0.1352 (0.0035)	0.2754 (0.0349)
	500	0.0765 (0.0009)	0.0447 (0.0011)	0.0922 (0.0034)	0.2021 (0.0155)
	1000	0.0571 (0.0016)	0.0349 (0.0014)	0.0735 (0.0021)	0.1887 (0.0058)
optical	10	0.3796 (0.0830)	0.2910 (0.0396)	0.3668 (0.0642)	0.3146 (0.0292)
	25	0.2402 (0.0293)	0.2140 (0.0155)	0.2540 (0.0197)	0.2454 (0.0321)
	50	0.2152 (0.0155)	0.1885 (0.0147)	0.2184 (0.0194)	0.2157 (0.0202)
	100	0.1652 (0.0088)	0.1642 (0.0085)	0.1716 (0.0085)	0.1693 (0.0128)
	200	0.1229 (0.0087)	0.1258 (0.0059)	0.1369 (0.0020)	0.1252 (0.0077)
	500	0.0600 (0.0034)	0.0623 (0.0023)	0.0762 (0.0034)	0.0611 (0.0039)
	1000	0.0367 (0.0020)	0.0374 (0.0015)	0.0582 (0.0022)	0.0363 (0.0016)
Pipe	10	0.1039 (0.0057)	0.0877 (0.0057)	0.1025 (0.0040)	0.0903 (0.0058)
	25	0.0884 (0.0022)	0.0602 (0.0052)	0.0869 (0.0019)	0.0601 (0.0031)
	50	0.0738 (0.0017)	0.0438 (0.0024)	0.0744 (0.0025)	0.0473 (0.0022)
	100	0.0598 (0.0037)	0.0316 (0.0014)	0.0589 (0.0036)	0.0351 (0.0015)
	200	0.0434 (0.0025)	0.0244 (0.0019)	0.0439 (0.0025)	0.0254 (0.0007)
	500	0.0325 (0.0013)	0.0160 (0.0014)	0.0337 (0.0012)	0.0148 (0.0007)
	1000	0.0261 (0.0006)	0.0125 (0.0004)	0.0274 (0.0009)	0.0108 (0.0003)

Table 17: **What the generic corrector adds to the structured branch.** Each entry is PRIOR-STRUCT$_{\text{no-free}}$/PRIOR-STRUCT, the error with the free corrector removed over the error with it; a value near 1 means the corrector contributes nothing and the structured branch carries the result. The parameter column is PRIOR-STRUCT against the same model with the corrector deleted, and the last column repeats the PRIOR-RAND/PRIOR-STRUCT ratio of Table 3.

	budget N					at $N = 200$	
Benchmark	10	25	50	100	200	params	rand/struct
2D Poisson	0.20	1.07	1.27	1.23	1.31	220,558 → 258	7.18×
3D layered	1.00	1.01	1.02	1.04	1.01	235,374 → 162,024	7.58×
annulus	1.01	0.96	0.89	0.98	0.97	38,468 → 16,368	22.21×
elasticity	1.00	0.99	1.00	0.99	0.98	59,914 → 31,414	2.16×
optical	0.74	1.00	1.12	1.28	1.67	21,916 → 66	1.09×
Pipe	1.00	1.03	1.01	1.01	0.98	859,680 → 12,130	1.80×

fivefold at N=10 (0.0032 against 0.0162) and costs 1.2–1.3× from N=50 on, so the structured branch is not merely sufficient there, it is better left alone when labels are scarcest.

Optical is the exception and it is the informative one. Without the corrector the model is about 0.21 at every budget from N=10 to N=200, flat and worse than PRIOR at 0.1229, so on that benchmark

the corrector, not the basis, produces the result. This is the same benchmark on which Table 3 resolves no difference between PRIOR-STRUCT, PRIOR and PRIOR-POD: the construction there returns a first-order sensitivity of the prior rather than a basis the residual actually lies in, and the flatness is what an inadequate basis looks like.

The parameter counts sharpen the reading. The ablated model on 2D Poisson has 258 parameters against the 220,300 of PRIOR and is five times more accurate at $N{=}200$ (0.0033 against 0.0161); on optical it has 66 and is worse than the prior it extends. Where the construction exposes the right subspace, the dimension of what remains to be learned is small enough to count; where it does not, no amount of structure substitutes for a learner.

We report PRIOR-STRUCT with the corrector throughout the main text, because that is the model every reported number was produced by and because it is the form that does not require knowing in advance whether $\Phi_{\mathcal{F}_0}$ is adequate. The ablation says what that choice costs where the basis is good.

G DATA-DERIVED PRIORS

All three peers are fitted, for every seed, to the same N training cases as the prior, and receive the 34 standardised inputs of Appendix E: the free-stream velocity, the angle of attack and the 32 surface ordinates. None is a neural network. The **reduced basis** (Sirovich, 1987) keeps the leading $\min(20, N-1)$ POD modes of the centred training fields and maps the input to their coefficients by ridge regression with intercept, the weight chosen from $10^{-6}, \ldots, 10^{4}$ by three-fold cross-validation on the training cases. **Retrieval** (Yue et al., 2025) predicts a Gaussian-kernel weighted average of the training fields in input space, the bandwidth chosen from $\{0.1, 0.25, 0.5, 1, 2, 4\}$ times the median pairwise training distance by leave-one-out error on the training cases. The **learned operator** maps 256 random Fourier features of the input, with frequencies scaled by the inverse of the same median distance, to the same POD coefficients by the same cross-validated ridge regression, a small kernel surrogate for a learned operator.

Two provisions favour the peers. They see richer inputs than the prior, which receives only the dynamic pressure and the angle; and the peer reported for each seed is the best of the three on the held-out error, selected after the fact. Appendix H places data-derived anchors, defined there, in the anchor slot of the same residual learner on nine benchmarks.

H ANCHOR ABLATION ACROSS BENCHMARKS

Appendix G compares the equation-derived prior with data-derived priors on AIRFRANS alone. This appendix repeats the comparison on nine benchmarks as an ablation of one slot of Eq. 10: the anchor u_0 is swapped and everything else is held fixed. The corrector is the PRIOR corrector throughout, a zero-initialised DeepONet (Appendix C.2) trained on $u - u_0$ with the optimisation of Appendix C.2, so the columns differ only in what is placed in front of it.

Anchors. *None*, $u_0 = 0$, is RAW and is not repeated in the table. *Equation*, $u_0 = u_p$, is PRIOR. *Retrieval* (Yue et al., 2025) returns the inverse-distance weighted mean of the five stored solutions nearest in standardised input space, the query instance excluded; with the learned residual on top this is the DELTAPHI construction. *Reduced basis* (Sirovich, 1987) keeps the eight leading left singular vectors of the centred stored fields and maps the input to the coefficients by least squares. *RFF* is the learned operator of Appendix G: 256 random Fourier features of the input mapped to twenty reduced-basis coefficients by cross-validated ridge regression. *DeepONet* is the anchor of Ma et al. (2026a): an operator of the same class and budget as the corrector, trained on the same instances with standardised targets, that is, RAW fitted to them. *Mean field* is the mean of the stored fields. Retrieval and RB here are simpler than the peers of the same names in Appendix G (five nearest neighbours rather than a kernel, eight modes and least squares rather than twenty and cross-validated ridge), and the AIRFRANS row uses these definitions.

Information. Every data-derived anchor is fitted, on each seed, to the same N training instances the corrector sees, so the comparison is at equal information and the equation anchor is the only one consuming no labels. AIRFRANS is the exception noted in Section 4.3: its prior is itself fitted on those instances, so there both sides use them.

Table 18: **Swapping the anchor, everything else held fixed.** Relative L_2 error, mean over five seeds; bold marks the row minimum. The anchors are those listed above, every data-derived one fitted to the same N instances the corrector sees. RB diverges on optical at N=200 and is marked †.

Benchmark	N	equation (u_p)	Retrieval	RB	RFF	DeepONet	Mean
1D Darcy	10	0.1490	0.1815	0.3023	0.1624	0.1690	**0.1437**
	200	0.0328	0.0614	0.2179	0.0351	**0.0268**	0.0276
2D Darcy	10	0.1607	0.1613	**0.1499**	0.1593	0.3519	0.1744
	200	**0.0821**	0.1091	0.1009	0.0875	0.1104	0.0866
2D Poisson	10	**0.1169**	0.6593	0.4467	0.5715	0.7007	0.6941
	200	**0.0161**	0.3817	0.0745	0.2023	0.2214	0.2117
3D Darcy	10	0.1792	**0.1702**	0.1704	0.1774	0.5290	0.1865
	200	0.1235	0.1252	**0.0994**	0.1215	0.2005	0.1227
3D layered	10	**0.1908**	1.5504	4.0656	1.6028	1.1418	1.6015
	200	**0.2377**	1.8025	4.5239	1.9351	1.6233	1.5872
AirfRANS	10	**0.3958**	1.5889	1.7960	1.2101	1.3851	1.2671
	200	**0.0507**	0.3678	0.3395	0.1112	0.0814	0.0896
Annulus	10	**0.6866**	1.0388	0.9106	0.8861	0.9974	0.9939
	200	**0.5195**	0.8118	0.7117	0.6320	0.6353	0.5765
Elasticity	10	**0.2749**	0.8947	0.8360	0.8791	0.9113	0.9201
	200	**0.1098**	0.6551	0.5878	0.4815	0.5211	0.5178
Optical	10	0.3796	0.5520	0.4964	**0.3655**	0.6052	0.4827
	200	**0.1229**	0.1899	100.47†	0.1418	0.1479	0.1718

Result. Table 18 reports nine benchmarks at $N = 10$ and $N = 200$. The equation anchor is the most accurate of the six on twelve of the eighteen rows, and on every row of 2D Poisson and of the four benchmarks whose input enters through the geometry or the layout (3D layered, annulus, elasticity and AIRFRANS), where the margin is largest: at $N = 200$ it is 0.0161 against 0.0745 for the best data-derived anchor on 2D Poisson, 0.2377 against 1.5872 on 3D layered, and 0.1098 against 0.4815 on elasticity. Over the nine benchmarks at $N = 200$ the data anchors are 2.2 to 3.6× the equation anchor on geometric mean (retrieval 3.3, RB 3.6 excluding its divergence on optical, RFF 2.3, DeepONet 2.3, mean field 2.2).

Five of the six rows it loses are in the Darcy family, where the effective-medium construction leaves $\varepsilon_{\text{prior}} \approx 0.19$–$0.22$ and the equation anchor stays within $1.25\times$ of the best column: a learned DeepONet anchor leads on 1D Darcy at $N = 200$ (0.0268 against 0.0328), a reduced basis on 3D Darcy (0.0994 against 0.1235), and the differences at $N = 10$ are within the seed spread. This is the same boundary as Table 2: where the construction resolves little, what the equation supplies is replaceable by labels; where it resolves the part of the map that data cannot cheaply recover, it is not.

A learned anchor is not a substitute. The DeepONet anchor is RAW of the same class and budget as the corrector, fitted to the same instances, and it is the arrangement of Ma et al. (2026a). It leads on one of the eighteen rows and is $2.3\times$ the equation anchor on geometric mean at $N = 200$: stacking a second learner in front of the corrector adds parameters and optimisation, not information.

I HELD-OUT DISTRIBUTION SHIFT

The two equation-derived objects are recomputed for every instance: u_p and $\Phi_{\mathcal{F}_0}$ are solves of $\mathcal{F}_0$ on the instance at hand, so they are as correct on an input the training set never covered as on one it did. RAW learns the whole map from data, and PRIOR-POD builds its basis from the training residuals; both are tied to the distribution they were fitted on. This appendix measures whether that distinction shows up in the error.

Protocol. Except on Pipe, training is unchanged (same instances, seeds, budget and optimiser) and only the evaluation set is added, so the in-distribution column is bit-identical to the corresponding cells of Tables 3 and 11; Pipe is retrained on the centre of its pool, as described below, and its in-distribution row differs accordingly. For the benchmarks whose instance splits into load and geometry (2D Poisson excepted), we hold the source, boundary or load data fixed at the values of the in-distribution evaluation set and redraw the geometry alone from a widened distribution, keeping by rejection only the draws that fall outside the coordinate range the training set spans;

the shifted set is therefore pure extrapolation in the domain of β_θ. For 2D Poisson the geometry is fixed and the shift is applied to the input function, whose Gaussian random field is made rougher ($\alpha = 2 \to 1.5$) at matched root-mean-square. For Pipe and optical the same rejection is instead applied to the first principal component of the wall profile and of the design profile; Pipe has fixed measured data, so instead of redrawing shapes we split its pool by that coordinate and train only on the centre.

Table 19: **Error under distribution shift (N=200, DeepONet).** Relative L_2 error, mean over five seeds. "in" is the standard evaluation set, "shifted" the held-out set described above; bold marks the row minimum on the shifted set.

Benchmark	set	RAW	PRIOR	PRIOR-STRUCT	PRIOR-RAND	PRIOR-POD
2D Poisson	in	0.2240	0.0161	0.0025	0.0179	0.0134
	shifted	0.2779	0.0196	**0.0050**	0.0225	0.0164
3D layered	in	1.6102	0.2377	0.0347	0.2633	0.2253
	shifted	2.8965	0.3988	**0.1343**	0.7795	0.8390
annulus	in	0.5774	0.5195	0.0251	0.5568	0.5738
	shifted	2.5677	3.0885	**0.4065**	4.4736	4.1016
elasticity	in	0.5223	0.1098	0.0626	0.1352	0.2754
	shifted	0.5635	**0.1261**	0.1335	0.1811	0.3743
Pipe	in	0.1065	0.0435	0.0239	0.0429	0.0251
	shifted	0.1330	0.0452	**0.0236**	0.0459	0.0263
optical	in	0.1605	0.1229	0.1258	0.1369	0.1252
	shifted	0.3929	0.3339	**0.3164**	0.3252	0.3307

Result. On the shifted set (Table 19) PRIOR-STRUCT is the most accurate formulation on five of the six benchmarks, and on Pipe its error does not move at all (0.0236 against 0.0239) while RAW degrades by $1.25\times$. The margin is largest where the basis carries the geometry: on annulus every other formulation exceeds an error of one, which is what predicting zero would give, while PRIOR-STRUCT stays at 0.41; on 3D layered it is 0.13 against 0.40 for the next best.

These comparisons are of levels, not of ratios. Measured as a ratio to its own in-distribution error, PRIOR-STRUCT degrades the most ($16\times$ on annulus against $4.5\times$ for RAW), but only because its in-distribution error is the smallest number in the table. The claim the ratio would support, that the structured branch is the most fragile formulation, is contradicted by every absolute comparison in the same rows.

Elasticity is the exception: there PRIOR is ahead of PRIOR-STRUCT on the shifted set (0.1261 against 0.1335), and it stays ahead at every budget (0.0704 against 0.0771 at N=1000). The basis itself still transfers (PRIOR-STRUCT remains ahead of PRIOR-RAND and PRIOR-POD), so what fails is the coefficient map. Elasticity is the one benchmark whose basis does not separate by mode, so β_θ must learn a full 64×8 coupling from three shape coefficients (Appendix C.5); the failure is consistent with it having the most to learn and the least reason to extrapolate. The distinction this appendix draws is therefore between the two equation-derived objects: u_p and $\Phi_{\mathcal{F}_0}$ are computed per instance and transfer by construction, while β_θ is learned and transfers only as far as its training range.

J MATCHED BASIS WIDTH AND POD-DEEPONET

Two asymmetries could account for the margin of PRIOR-STRUCT over PRIOR-POD in Table 3. First, $\Phi_{\mathcal{F}_0}$ is truncated by the construction (on elasticity the Laurent orders are cut to $|k| \leq 8$ and the load modes to $j \leq 2$, giving $n_b = 512$ instead of 4096; Appendix C.5), while PRIOR-POD is given every singular vector the training residuals admit, which at N=200 is $\min(200, n_b)$ of them; the usual practice is to keep far fewer. Second, PRIOR-POD is a control inside our own architecture, not POD-DeepONet (Lu et al., 2022), which is a different model and tunes its mode count per problem. This appendix removes both asymmetries.

Matched width. We cut all three bases to the same k columns and refit: PRIOR-POD keeps the leading k singular vectors, and $\Phi_{\mathcal{F}_0}$ and the random basis keep the k columns of largest energy on the training set, so the three arms differ only in where the basis came from.

Table 20: **All three bases cut to the same width (N=200, DeepONet).** Relative L_2 error, mean over five seeds. "full" is the width used in the main text, given in the last column. Bold marks the best arm at each width.

Benchmark	basis	k=8	k=16	k=32	k=64	full	full n_b
2D Poisson	PRIOR-STRUCT	**0.0038**	**0.0030**	**0.0026**	**0.0025**	**0.0025**	256
	PRIOR-RAND	0.0170	0.0168	0.0170	0.0171	0.0179	
	PRIOR-POD	0.0128	0.0127	0.0128	0.0132	0.0134	
3D layered	PRIOR-STRUCT	**0.1988**	**0.1722**	**0.1378**	**0.1025**	**0.0347**	3072
	PRIOR-RAND	0.2402	0.2397	0.2376	0.2399	0.2633	
	PRIOR-POD	0.2108	0.2100	0.2116	0.2153	0.2253	
annulus	PRIOR-STRUCT	**0.4571**	**0.3860**	**0.2835**	**0.1736**	**0.0251**	216
	PRIOR-RAND	0.5322	0.5278	0.5244	0.5321	0.5568	
	PRIOR-POD	0.5706	0.5708	0.5678	0.5756	0.5738	
elasticity	PRIOR-STRUCT	0.1288	0.1368	0.1654	0.1840	**0.0626**	512
	PRIOR-RAND	**0.1102**	**0.1111**	**0.1143**	**0.1145**	0.1352	
	PRIOR-POD	0.3271	0.3185	0.3005	0.2917	0.2754	
optical	PRIOR-STRUCT	**0.1221**	0.1317	0.1388	0.1258	0.1258	64
	PRIOR-RAND	0.1232	0.1253	0.1283	0.1369	0.1369	
	PRIOR-POD	0.1298	**0.1226**	**0.1267**	**0.1252**	**0.1252**	
Pipe	PRIOR-STRUCT	0.0440	0.0436	0.0438	0.0436	**0.0244**	128
	PRIOR-RAND	0.0449	0.0437	0.0445	0.0449	0.0439	
	PRIOR-POD	**0.0307**	**0.0289**	**0.0281**	**0.0268**	0.0254	

The width was not what separated the arms (Table 20). PRIOR-POD gains little from narrowing: its best width is within 7% of the full-width number on every benchmark, so narrowing it to the conventional range changes no ordering in Table 3. On 2D Poisson, 3D layered and annulus the equation-derived basis is ahead at every k, and the margin grows with k rather than with the asymmetry: from $3.4\times$, $1.06\times$ and $1.16\times$ at k=8 to $5.3\times$, $2.1\times$ and $3.1\times$ at k=64, against the better of the two controls. The full-width numbers of Table 3 continue that trend.

Elasticity is the exception, and it runs the other way: truncating $\Phi_{\mathcal{F}_0}$ by training-set energy makes PRIOR-STRUCT *worse* as k grows (0.1288 at k=8 to 0.1840 at k=64) and leaves it behind PRIOR-RAND at every k, while the full 512-column basis reaches 0.0626. The Laurent basis there is not ordered by training-set energy (the fields a given shape needs are not the fields that dominate the pooled residual), so an energy-ranked cut keeps the wrong columns. Pipe fails for the same reason: 52 of its 64 highest-energy columns have $m > 16$, so at $k \leq 64$ PRIOR-STRUCT stays at the prior (0.0436–0.0440 against 0.0434), while the full 128-column basis reaches 0.0244. This is a statement about how to truncate, not about how wide to be: the construction's own truncation, by Laurent order on elasticity and by (m, n) on Pipe, is what the main text uses.

POD-DeepONet. The PRIOR-POD arm is not the model of Lu et al. (2022): it replaces the basis inside our decomposition, keeping u_p and the corrector. POD-DeepONet has no prior, sets the trunk to the leading p POD modes of the training *solutions* with $\bar{y}$ as the zeroth mode, and lets the branch emit the p coefficients. We run it directly, sweeping p over the range that paper uses (Table 21).

Table 21: **POD-DeepONet (Lu et al., 2022) against structural injection (N=200).** Relative L_2 error, mean over five seeds, p the number of POD modes in the trunk; p=115 is the largest value Lu et al. (2022) use (piecewise-constant Darcy). The last two columns repeat PRIOR-STRUCT and u_p alone from Table 3.

Benchmark	p=8	p=16	p=32	p=64	p=115	PRIOR-STRUCT	u_p alone
2D Poisson	0.2499	0.2494	0.2176	0.1901	0.1779	**0.0025**	0.1827
3D layered	1.2927	1.3809	1.3570	1.3912	1.3003	**0.0347**	0.1751
annulus	0.6639	0.6478	0.6646	0.6401	0.6223	**0.0251**	0.6649
elasticity	0.5371	0.4850	0.5532	0.5441	0.5275	**0.0626**	0.3214
Pipe	0.0416	0.0320	0.0303	0.0293	0.0263	**0.0244**	0.1070
optical	0.1362	**0.1049**	0.1099	0.1127	0.1175	0.1258	0.2941

At its best p, POD-DeepONet is 7.7 to $71\times$ above PRIOR-STRUCT on 2D Poisson, 3D layered, annulus and elasticity, and $1.08\times$ above it on Pipe. On optical at N=200 it is the most accurate model in this paper: 0.1049 at p=16, against 0.1258 for PRIOR-STRUCT and 0.1229 for the prior alone with a corrector. Optical is the benchmark whose construction returns a first-order sensitivity rather than a solution basis (Appendix B.5), and the one on which PRIOR-STRUCT, PRIOR and

PRIOR-POD do not separate in Table 3; that a data-derived trunk does better there is consistent with the condition the main text states rather than an exception to it.

K RELATION TO RESIDUAL AND REFERENCE-BASED METHODS

In the factorization of Eq. 10, the closest prior methods fill the same slots with learned or simulated objects. DELTAPHI (Yue et al., 2025) fills the anchor with a retrieved training solution, $u_0 = u_{\mathrm{ref}}$, and learns $u - u_{\mathrm{ref}}$; the retrieval peer of Appendix G is this construction with a kernel weighting, and Table 15 compares the best such peer with the equation-derived prior on AIRFRANS; Appendix H places retrieval in the anchor slot of the same residual learner. Ma et al. (2026a) fill the anchor with a first DeepONet and the objective with the misspecified residual, then correct with a second operator; our PRIOR formulation differs in that the anchor is solved rather than learned. Appendix H places a learned DeepONet anchor in front of our corrector on nine benchmarks at $N{=}200$: it is the more accurate of the two on one of them (1D Darcy, 0.0268 against 0.0328) and worse on the other eight, by $4.7\times$ on elasticity and $6.8\times$ on 3D layered, for a geometric mean of $2.3\times$ the solved anchor. The other four data-derived anchors behave the same way (2.2–$3.6\times$). Ma et al. (2026b) fill no slot: they simulate simplified equations as auxiliary training data, whereas we solve them once and never train on them. Dang et al. (2026) fill the query-side factor with interface locations predicted by an auxiliary network, whereas the layered construction supplies $b_j(a)$ in closed form.

POD-DeepONet (Lu et al., 2022) also fixes the trunk, with the leading POD modes of the training fields. In the notation of Section 3 it and PRIOR-STRUCT read

$$\begin{aligned} &\text{POD-DeepONet:} \quad \hat{u}_\theta(x;a) = \bar{u}(x) + \sum_{k=1}^{p} b_{\theta,k}(a)\, \phi_k^{\mathrm{POD}}(x), \\ &\text{PRIOR-STRUCT:} \quad \hat{u}_\theta(x;a) = u_p(x;a) + \sum_{k=1}^{n_b} \beta_{\theta,k}(a_g)\, F_k(a_f)\, S_k(x;a_g) + \epsilon_\theta(x;a), \end{aligned} \tag{35}$$

where $\bar{u}$ is the mean of the N training fields and b_θ a branch network. In Eq. 10, POD-DeepONet fills the anchor with the mean field and the query-side factor with a snapshot basis. The anchor $\bar{u}$ and the modes ϕ_k^{POD} are extracted from labels and are the same for every instance. The anchor u_p and the basis $\Phi_{\mathcal{F}_0}$ are constructed from $\mathcal{F}_0$ without labels, and the basis changes with the instance, through the source factor and, where the geometry varies, through $S_k(x;a_g)$. The POD modes span the solution, whereas $\Phi_{\mathcal{F}_0}$ is built for the residual. Two further differences follow. A POD basis has at most $\min(N, n_q)$ informative columns, whereas the width of $\Phi_{\mathcal{F}_0}$ is fixed by the construction; and POD modes are defined on the grid of the snapshots, whereas $\Phi_{\mathcal{F}_0}$ is a function of x and can be evaluated on any query set. PRIOR-POD isolates where the spatial factor comes from: it keeps u_p and F and replaces S by the leading POD modes of the training residuals, label-derived and shared by every instance. The data-derived ingredients of POD-DeepONet already enter Appendix H: the mean-field anchor is its anchor, the reduced-basis anchor is its basis with a linear map to the coefficients, and the RFF anchor is its basis with a kernel map. The reduced basis leads only on two Darcy rows (2D Darcy at $N{=}10$, 3D Darcy at $N{=}200$), where the effective-medium prior itself is weak; the equation-derived anchor is stronger where labels are few and the geometry varies across instances. PRIOR-POD trails PRIOR-STRUCT at every budget on the four benchmarks whose residual carries its own structure (on 2D Poisson 0.0687 against 0.0162 at $N{=}10$ and 0.0088 against 0.0023 at $N{=}1000$), and matches or passes it only on Pipe and optical (Section 4.1).

L ADDITIONAL EXPERIMENTAL DETAIL

Cost of the prior against the physics-informed loss. The prior and the residual loss pay for the equation on opposite schedules, and we time both against training. Solving $\mathcal{F}_0$ once per instance, including the interpolation of the sensor readings that the prior and the residual loss both consume, costs a median of 0.05% (at most 1.1%, on annulus) of one training run of PRIOR over the entire test set, and is paid once per instance rather than once per optimization step. The residual loss is the opposite: applying the discretized operator $\mathcal{F}_0(\hat{u};a)$ on the collocation grid at every step makes one PI run cost 1.0–$2.1\times$ (median $1.2\times$) one PRIOR run on the same benchmark, each timed alone on one GPU, and the oracle runs it once per point of the nine-point λ grid, 9–$19\times$ one PRIOR run per

cell. Structural injection adds no further solve, since $\Phi_{\mathcal{F}_0}$ is a by-product of the same construction. Against the reference solver, where it differs in kind from the construction, the construction is $239\times$ and $628\times$ cheaper on 2D and 3D Darcy and $2.2\times$ on elasticity (one CPU thread, mean over ten test instances).