# DIRECT LEARNING OF CALIBRATION-AWARE UNCERTAINTY FOR NEURAL PDE SURROGATES

**Carlos Stein Brito**
NightCity Labs, Lisbon, Portugal
carlos.stein@nightcitylabs.ai

## ABSTRACT

Neural PDE surrogates are often deployed in data-limited or partially observed regimes where downstream decisions depend on calibrated uncertainty in addition to low prediction error. Existing approaches obtain uncertainty through ensemble replication, fixed stochastic noise such as dropout, or post hoc calibration. Cross-regularized uncertainty learns uncertainty parameters during training using gradients routed through a held-out regularization split. The predictor is optimized on the training split for fit, while low-dimensional uncertainty controls are optimized on the regularization split to reduce train–test mismatch, yielding regime-adaptive uncertainty without per-regime noise tuning. The framework can learn continuous noise levels at the output head, within hidden features, or within operator-specific components such as spectral modes. We instantiate the approach in Fourier Neural Operators and evaluate on APEBench sweeps over observed fraction and training-set size. Across these sweeps, the learned predictive distributions are better calibrated on held-out splits and the resulting uncertainty fields concentrate in high-error regions in one-step spatial diagnostics.

## 1 INTRODUCTION

Neural operators provide a practical route to fast PDE surrogate modeling and have become a standard backbone for data-driven scientific simulation (Li et al., 2021; Lu et al., 2021; Takamoto et al., 2022). In many downstream settings, however, raw predictive accuracy is not sufficient: decisions require uncertainty estimates that are both numerically stable and calibration-aware under distribution shift.

The dominant baselines are MC dropout, deep ensembles (Gal & Ghahramani, 2016; Lakshminarayanan et al., 2017), and post hoc conformal wrappers (Angelopoulos & Bates, 2021; Ma et al., 2024). While these approaches provide essential baselines, they lack a direct mechanism to optimize uncertainty against the train–test mismatch driving overfitting in neural PDE pipelines. Consequently, fixed stochastic settings often fail to adapt to regime-specific scarcity or sparsity.

We formulate uncertainty as a learnable optimization target coupled directly to the regularization objective. Concretely, we keep a predictive uncertainty branch optimized on the training split through the likelihood objective, and we introduce a separate generalization-uncertainty branch whose parameters are updated **only** using gradients computed on a regularization split. This brings the cross-regularization (XReg) principle (Brito, 2025) to neural PDE surrogates, shifting the focus from architectural noise placement to objective-routed learning. Distinct from conventional hyperparameter tuning, XReg updates uncertainty parameters online using the regularization set without re-fitting the predictor backbone. Throughout, we use "predictive" and "generalization" uncertainty as operational labels defined by this gradient routing rather than as a physical decomposition.

Our contributions are:

- **Cross-regularized uncertainty learning for neural PDE surrogates.** We partition parameters into predictor and uncertainty controls and train them with a train/reg split: predictor parameters

are updated on the training split, while generalization-uncertainty controls are updated online from regularization-split gradients (Algorithm 1).

- **Regime-adaptive uncertainty scaling from held-out gradients.** The regularization update directly adjusts uncertainty magnitude in response to train–test mismatch estimated on the regularization split, yielding uncertainty that increases when observations are sparse or data are scarce and decreases when supervision is informative.
- **Empirical calibration and localization evidence.** On APEBench sweeps over observed fraction and training-set size, the learned predictive distributions improve held-out Gaussian-mixture calibration relative to MC dropout and 3-member ensembles under matched protocols, and one-step spatial diagnostics show uncertainty concentrating in high-error regions.

## 2 Related Work

**Neural operator surrogates for PDEs.** Operator-learning architectures such as DeepONet and Fourier Neural Operators are now standard baselines for PDE surrogate modeling (Lu et al., 2021; Li et al., 2021). Benchmark efforts have further clarified both their strengths and their calibration/generalization challenges across tasks and resolutions (Takamoto et al., 2022).

**Uncertainty baselines in deep models.** Widely used uncertainty baselines include MC dropout and deep ensembles (Gal & Ghahramani, 2016; Lakshminarayanan et al., 2017). Conformal approaches provide finite-sample calibration guarantees and have recently been adapted to operator-learning settings (Angelopoulos & Bates, 2021; Ma et al., 2024). Our focus is an adaptive coupling between train and reg. set objectives that tunes uncertainty scale to regime-specific mismatch.

**Validation-gradient complexity control.** Prior work on cross-regularization (XReg) introduces learning regularization controls directly from held-out gradients (Brito, 2025). We use this principle with neural operators to disentangle predictive and generalization uncertainty within a single optimization loop.

## 3 XReg Primer

Cross-regularization (XReg) (Brito, 2025) separates feature fitting from complexity control by routing their gradients through different data partitions. We decompose uncertainty into two operational categories: *predictive uncertainty*, which absorbs residuals required for training fit, and *generalization uncertainty*, which smooths the model to reduce train–held-out mismatch. This decomposition is model-relative rather than mechanistic: predictive uncertainty mirrors aleatoric uncertainty in absorbing residual variability (including approximation and optimization error), while generalization uncertainty captures epistemic ambiguity driven by limited evidence. The distinction between predictive and generalization uncertainty arises solely from the optimization signal rather than distinct probabilistic architectures.

### 3.1 Cross-Regularized Optimization Split

Let model parameters be partitioned into deterministic backbone parameters $\theta$, predictive-noise parameters $\psi$, and generalization-noise parameters $\rho$. XReg uses alternating updates:

$$(\theta, \psi)_{t+1} = (\theta, \psi)_t - \eta_{\theta,\psi} \nabla_{(\theta,\psi)} \mathcal{L}_{\text{train}}((\theta, \psi)_t, \rho_t), \tag{1}$$

$$\rho_{t+1} = \rho_t - \eta_\rho \nabla_\rho \mathcal{L}_{\text{reg}}((\theta, \psi)_{t+1}, \rho_t). \tag{2}$$

## 4 Cross-Regularized Uncertainty Framework

### 4.1 Backbone-Agnostic Formulation

Consider a generic neural operator backbone $f_\theta$. We augment it with two stochastic parameter sets:

$$\psi \in \Psi \quad \text{(predictive-noise parameters)}, \qquad \rho \in \mathcal{R} \quad \text{(generalization-noise parameters)}.$$

For input $x$, the model produces predictive moments

$$\mu_{\theta,\psi,\rho}(x), \qquad \sigma_{\text{pred},\psi}(x),$$

where $\rho$ may affect the output distribution through stochastic perturbations of internal features and/or output layers.

### 4.2 Where Noise Can Be Injected

The framework is defined by the train/regularization optimization split, so the uncertainty pathways can be attached to many model sites. In this paper we focus on output-head and internal-feature instantiations, with the option to co-locate predictive and generalization noise while keeping them distinct through gradient routing.

For the internal variants used here, stochasticity is implemented as multiplicative Gaussian perturbations of selected latent feature blocks and, when enabled, selected Fourier-mode channels, using $h \mapsto h \odot (1 + \sigma\epsilon)$. This placement flexibility does not require a Bayesian weight posterior and avoids committing to a single architectural proxy for aleatoric versus epistemic uncertainty.

### 4.3 Train and Regularization Objectives

**Train objective.** Train updates optimize a Monte Carlo approximation of the marginal (mixture) likelihood under generalization sampling:

$$\mathcal{L}_{\text{train}} = -\log \frac{1}{S} \sum_{s=1}^{S} p_{\text{hier}}(y \mid x, \omega_s),$$

with

$$p_{\text{hier}} = \exp\left( -\frac{1}{2} \left[ \log(2\pi\sigma_{\text{pred}}^2) + \frac{(y-\mu)^2 + \sigma_{\text{gen}}^2}{\sigma_{\text{pred}}^2} \right] \right).$$

In internal-noise variants, $\sigma_{\text{gen}}$ is implicit in sampled model instances, and the generalization contribution is measured through $\mathrm{Var}_\omega[\mu]$ at evaluation time. Train updates optimize $(\theta, \psi)$; regularization parameters $\rho$ are optimized only by $\mathcal{L}_{\text{reg}}$.

**Regularization objective.** Regularization updates are performed on held-out one-step pairs. We use two estimators in this work:

1. **Mixture NLL**:

$$\mathcal{L}_{\text{reg}}^{\text{mix}} = -\log \frac{1}{S} \sum_{s=1}^{S} p(y \mid \mu_s, \sigma_s^2),$$

   where each Monte Carlo sample induces one Gaussian.

2. **Moment-matched NLL**:

$$\mu_{\text{mm}} = \mathbb{E}[\mu_s], \quad \sigma_{\text{mm}}^2 = \mathbb{E}[\sigma_s^2 + \mu_s^2] - \mu_{\text{mm}}^2,$$
$$\mathcal{L}_{\text{reg}}^{\text{mm}} = -\log p(y \mid \mu_{\text{mm}}, \sigma_{\text{mm}}^2).$$

**Head-only special case (predictive and generalization branches at the same model position).** When both branches are output heads (without internal-layer stochastic sampling), the asymmetry is explicit:

$$\mathcal{L}_{\text{train}} = \frac{1}{2} \left[ \log(2\pi\sigma_{\text{pred}}^2) \; + \; \frac{(y-\mu)^2 + \sigma_{\text{gen}}^2}{\sigma_{\text{pred}}^2} \right],$$

whereas the regularization objective uses total variance

$$\sigma_{\text{tot}}^2 = \sigma_{\text{pred}}^2 + \sigma_{\text{gen}}^2, \qquad \mathcal{L}_{\text{reg}} = \frac{1}{2} \left[ \log(2\pi\sigma_{\text{tot}}^2) + \frac{(y-\mu)^2}{\sigma_{\text{tot}}^2} \right]$$

(or its MC-mixture counterpart when internal stochastic noise is enabled). This objective/data-routing asymmetry enables distinct learning dynamics even when predictive and generalization parameters are co-located at the output head.

**Algorithm 1: Cross-Regularized Dual-Noise Training**
**Input:** $\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{reg}}, T, k_{\text{reg}}, \eta_{\theta,\psi}, \eta_\rho, S$.
**Parameters:** backbone $\theta$, predictive-noise parameters $\psi$, generalization-noise parameters $\rho$.
**Output:** trained $(\theta, \psi, \rho)$.

**Step 1:** Initialize $(\theta, \psi, \rho)$.

**Step 2:** For $t = 1, \ldots, T$, repeat:

(a) Sample train mini-batch $B_{\text{train}} \sim \mathcal{D}_{\text{train}}$.
(b) Sample model instances $\omega_{1:S} \sim q_\rho$ and compute $\mathcal{L}_{\text{train}}(B_{\text{train}}; \theta, \psi, \omega_{1:S})$.
(c) Update
$$(\theta, \psi) \leftarrow (\theta, \psi) - \eta_{\theta,\psi} \nabla_{(\theta,\psi)} \mathcal{L}_{\text{train}}.$$
(d) If $t \bmod k_{\text{reg}} = 0$:
  (i) Sample regularization mini-batch $B_{\text{reg}} \sim \mathcal{D}_{\text{reg}}$.
  (ii) Sample $\omega_{1:S} \sim q_\rho$ and evaluate $\mathcal{L}_{\text{reg}}(B_{\text{reg}}; \theta, \psi, \rho, \omega_{1:S})$.
  (iii) Update
$$\rho \leftarrow \rho - \eta_\rho \nabla_\rho \mathcal{L}_{\text{reg}}.$$

**Step 3:** Return $(\theta, \psi, \rho)$.

Figure 1: Cross-regularized dual-noise training. Train updates optimize $(\theta, \psi)$ on $\mathcal{D}_{\text{train}}$, and periodic regularization updates optimize $\rho$ on $\mathcal{D}_{\text{reg}}$.

### 4.4 Algorithm

Every iteration updates $(\theta, \psi)$ on $\mathcal{D}_{\text{train}}$; every $k_{\text{reg}}$ steps, a second pass updates only $\rho$ on $\mathcal{D}_{\text{reg}}$. Here $q_\rho$ denotes the stochastic model family induced by $\rho$, and $\omega \sim q_\rho$ denotes one sampled model instance. In head-only variants, $\omega$ corresponds to sampled output-level generalization-noise realizations; in internal-noise variants, $\omega$ corresponds to multiplicative perturbations in selected latent feature blocks. In our main experiments, $k_{\text{reg}} = 5$; practical overhead and its dependence on $k_{\text{reg}}$ are summarized in Appendix A.7, and less frequent regularization updates approach near-standard training cost (Brito, 2025).

### 4.5 Instantiation in This Work

For empirical study, we instantiate the framework with a 1D Fourier Neural Operator backbone (Li et al., 2021). The model outputs

$$\mu_\theta(x_t), \quad \log \sigma_{\text{pred},\psi}(x_t), \quad \log \sigma_{\text{gen},\rho}(x_t),$$

with output-head and internal-layer variants used across ablations and sweeps.

### 4.6 Uncertainty Decomposition at Evaluation

At inference, we separate uncertainty into predictive and generalization terms:

$$U_{\text{pred}}(x) = \sigma^2_{\text{pred}}(x), \tag{3}$$
$$U_{\text{gen}}(x) = \mathrm{Var}_{\omega \sim q_\rho}\left[\mu(x; \omega)\right], \tag{4}$$
$$U_{\text{tot}}(x) \approx U_{\text{pred}}(x) + U_{\text{gen}}(x), \tag{5}$$

where $U_{\text{gen}}$ has two implementation cases used in this work:

$$\text{head-only: } U_{\text{gen}}(x) = \sigma^2_{\text{gen}}(x), \qquad \text{internal stochastic: } U_{\text{gen}}(x) = \mathrm{Var}_\omega[\mu(x; \omega)].$$

We report one-step likelihood and Gaussian-mixture calibration as the primary quantitative metrics, plus spatial localization diagnostics.

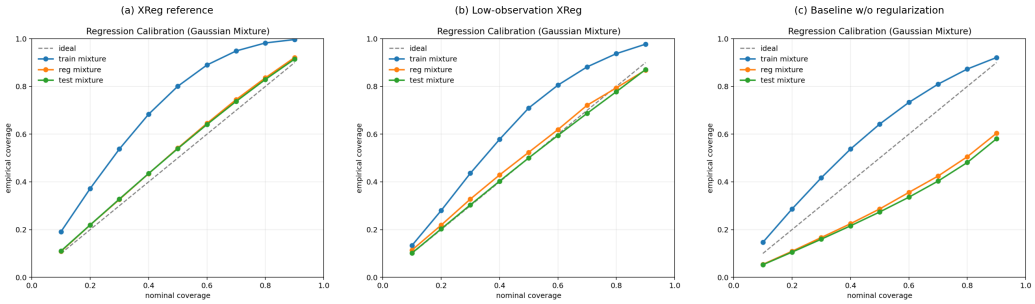



Figure 2: Regression-mixture calibration in three representative regimes. Left: XReg reference regime with moment-matched regularization. Middle: low-observation XReg regime ($30\%$ observed). Right: model without learned regularization. In all panels, XReg tracks the diagonal more closely on held-out splits, with the largest gains under limited observations.

# 5 Experimental Setup

## 5.1 Data and Pair Construction

The primary experiments use one-step teacher-forced pairs from APEBench trajectories, and the car experiment uses the OTNO pipeline on car-surface pressure data. Appendix A documents dataset provenance, split construction, masking protocol, and metric definitions.

## 5.2 Experimental Configurations Used in Figures

Main-paper figures compare four controlled regimes:

- **Low-observation XReg regime:** output-head predictive and generalization scales under partial observations.
- **Head-only decomposition ablation:** both uncertainty branches at the output head, used to test whether objective/data routing alone separates their roles.
- **Baseline without learned regularization:** same backbone family with no active learned generalization branch.
- **Internal-noise XReg reference regime:** internal generalization-noise injection enabled, predictive head enabled, head-level generalization branch disabled, and moment-matched regularization.

These regimes test decomposition identifiability, calibration behavior, and the effect of where uncertainty is injected.

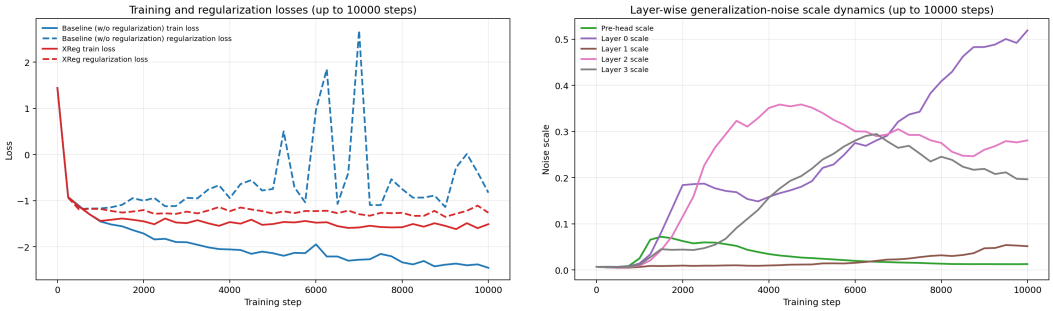
# 6 Empirical Calibration and Dynamics

## 6.1 Mixture Calibration Comparison

We prioritize calibration analysis to validate the method's ability to capture predictive distributions beyond simple mean error. Figure 2 shows representative regimes under a matched protocol; formal metric definitions (including $\mathrm{ECE}_{\mathrm{mix}}$ and one-step NLL normalization) are provided in Appendix A.6.

## 6.2 Optimization Dynamics and Learned Uncertainty Scales

Optimization trajectories (Figure 3) illustrate the coupling between train and regularization objectives and how uncertainty magnitude is distributed across injection sites.




(a) Train and regularization losses up to step 10,000.

(b) Layerwise evolution of generalization-noise scales up to step 10,000.

Figure 3: Optimization and uncertainty-allocation dynamics. Left: model without learned regularization versus XReg train/reg losses (solid=train, dashed=reg). XReg maintains a substantially tighter train-to-reg. set gap. Right: pre-head and layerwise generalization-noise scales. Uncertainty is allocated non-uniformly across layers, indicating learned structure.

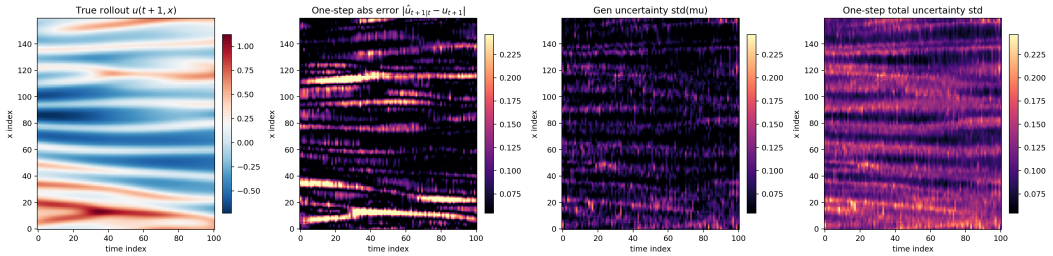



Figure 4: One-step teacher-forced maps (time indices 0 to 100): true field, absolute one-step error, generalization uncertainty ($\mathrm{Std}_\omega[\mu]$), and total predictive uncertainty. Error and uncertainty panels share the same numeric range (set by total-uncertainty limits) to enable direct spatial comparison. High-error regions co-localize with high predicted uncertainty, indicating spatially selective uncertainty assignment.

### 6.3 Spatially Resolved One-Step Uncertainty Tracking

To test spatial selectivity of uncertainty assignment, we analyze one-step spatiotemporal maps.

Figure 4 compares one-step error and predicted uncertainty at matched space–time coordinates. The uncertainty field varies spatially, concentrating in regions with larger residuals.

### 6.4 OTNO Car Surface-Pressure Experiment

Transfer beyond 1D trajectories is evaluated with a geometry-conditioned car-pressure experiment using an Optimal Transport Neural Operator (OTNO) backbone (Li et al., 2025). The data source and preprocessing provenance are documented in Appendix A.3.

The run shown in Figure 5 uses a deliberately data-limited split with $n_{\text{train}} = 30$ and $n_{\text{reg}} = 50$, with the remaining settings following the default protocol in Appendix A.1. Predictive uncertainty is parameterized at the output head, while generalization uncertainty is injected in internal layers.

Figure 5 shows three complementary views: absolute error on the car surface (left), cross-regularized uncertainty on the same surface (middle), and a roofline slice with mean and uncertainty bands (right). The spatial correspondence between the left and middle panels indicates that uncertainty concentrates in high-error regions and remains spatially selective. The slice view shows contributions from both predictive and generalization uncertainty, with uncertainty envelopes widening around difficult regions and narrowing where the fit is stable. Figure 8 reports additional uncertainty-scale trajectories for this run.

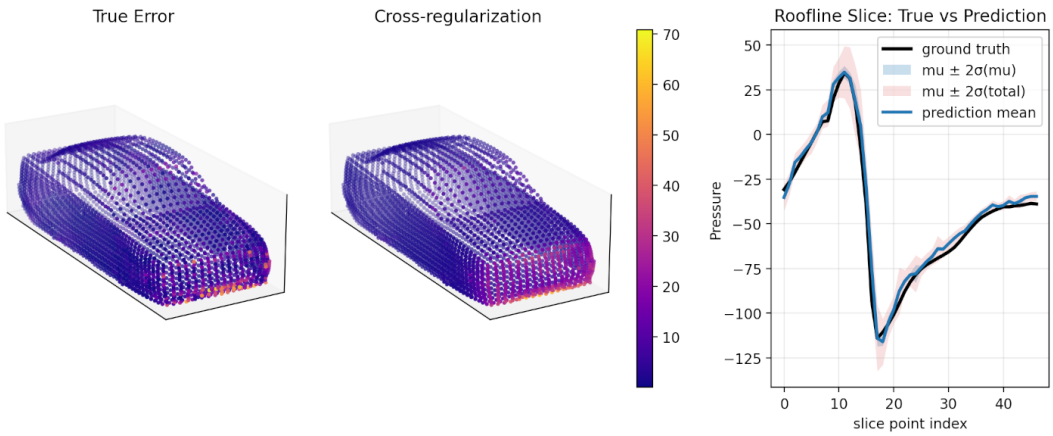



Figure 5: OTNO car-pressure experiment with cross-regularized uncertainty ($n_{\text{train}} = 30$, $n_{\text{reg}} = 50$). Left: absolute prediction error on the car surface. Middle: learned uncertainty field on the same sample. Right: roofline slice with ground truth, predictive mean, and uncertainty bands. Error hot spots and uncertainty hot spots are spatially aligned, and the slice shows calibrated widening of uncertainty in regions with larger residuals.

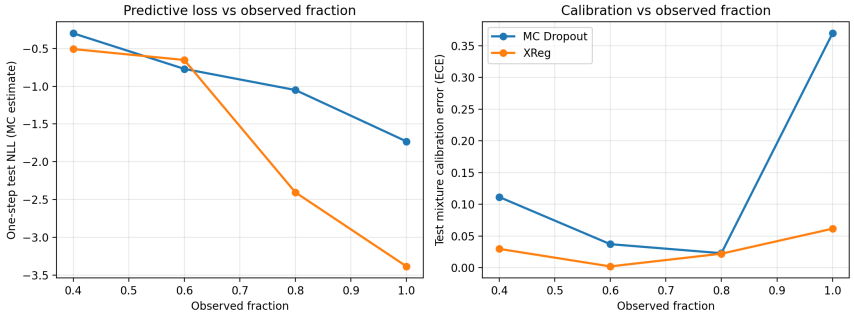



Figure 6: Observed-fraction sweep $(0.4, 0.6, 0.8, 1.0)$ comparing XReg and MC dropout on one-step test NLL (left, MC-evaluated) and test mixture ECE (right). The two-panel view separates fit quality from calibration quality. XReg is consistently better calibrated (lower ECE) across all observation fractions while preserving competitive one-step NLL.

### 6.5 Sweep Comparison: XReg vs. Baselines (Loss + ECE)

XReg is compared against two stochastic baselines under matched observation settings: MC dropout ($p = 0.1$) and deep ensembles (3 members). Dropout $p = 0.1$ is used as a compute-matched baseline; tuning $p$ per regime is possible but would undermine the no-hand-tuning objective of this comparison. Figure 6 compares performance scaling between XReg and MC dropout. Exact observed-fraction values are reported in Table 1, and the train-size sweep values are reported in Appendix Table 2. Deep-ensemble values are computed from the aggregated predictive mixture of the three trained members at each sweep point.

These results demonstrate that cross-regularized uncertainty scales are learned directly from regime-specific train–test mismatch. By construction, XReg updates generalization uncertainty from regularization-split gradients, allowing the uncertainty scale to track observation level and data regime during training. Relative to fixed-rate dropout and 3-member deep ensembles, this yields more consistent calibration across data and observability conditions.

| Obs. Frac | Method | Test NLL (MC)↓ | Test $\mathrm{ECE}_{\mathrm{mix}}$ ↓ | Reg $\mathrm{ECE}_{\mathrm{mix}}$ ↓ |
|---|---|---|---|---|
| 0.4 | XReg | **-0.5083** | **0.0296** | **0.0248** |
| 0.4 | MC Dropout | -0.2999 | 0.1115 | 0.1153 |
| 0.4 | Deep Ensemble (3) | -0.1002 | 0.1121 | 0.1125 |
| 0.6 | XReg | -0.6533 | **0.0020** | **0.0040** |
| 0.6 | MC Dropout | -0.7710 | 0.0370 | 0.0324 |
| 0.6 | Deep Ensemble (3) | **-0.8900** | 0.0491 | 0.0522 |
| 0.8 | XReg | **-2.4052** | **0.0220** | **0.0139** |
| 0.8 | MC Dropout | -1.0506 | 0.0228 | 0.0159 |
| 0.8 | Deep Ensemble (3) | -2.0772 | 0.0624 | 0.0530 |
| 1.0 | XReg | -3.3849 | **0.0614** | **0.0596** |
| 1.0 | MC Dropout | -1.7310 | 0.3696 | 0.3695 |
| 1.0 | Deep Ensemble (3) | **-3.5632** | 0.2943 | 0.2972 |

Table 1: Numeric comparison for the observed-fraction sweep. XReg achieves the lowest test and regularization mixture ECE across all four observation fractions, while maintaining competitive one-step likelihood.

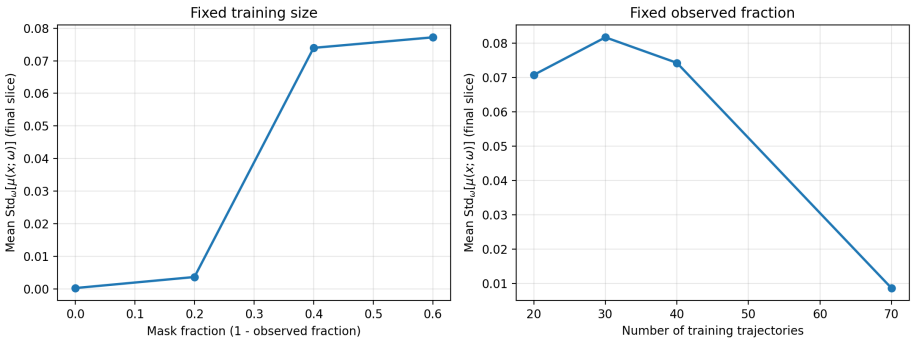



Figure 7: Noise/data scaling of learned generalization uncertainty, measured as mean $\mathrm{Std}_\omega[\mu]$ on the final slice. Left: increasing mask fraction (fixed train size $= 50$) corresponds to less observed signal and produces larger learned generalization uncertainty. Right: increasing train-set size (fixed observed fraction $= 0.7$) reduces learned generalization uncertainty. These trends indicate that XReg adapts uncertainty magnitude to regime difficulty.

### 6.6 Generalization Uncertainty Tracks Noise and Data Availability

Figure 7 isolates how the learned generalization uncertainty responds to regime difficulty. When the observed fraction decreases (weaker supervision), the learned generalization uncertainty increases; when the training-set size increases (more evidence), it decreases. Under the cross-regularized update, regularization-split gradients set the uncertainty scale in the direction predicted by train–test mismatch, producing regime-dependent noise levels.

### 6.7 Main Findings and Implications

Across matched protocols, cross-regularized uncertainty improves held-out mixture calibration while preserving competitive one-step likelihood, indicating that the calibration gains are not explained by trivial variance inflation. The supporting diagnostics are consistent with the proposed mechanism: optimization traces show a reduced train-to-regularization gap; learned uncertainty scales evolve non-uniformly across layers; and one-step spatial maps concentrate uncertainty in regions that exhibit larger one-step error.

## 7 Discussion: Model-Relative Uncertainty

Uncertainty in neural PDE surrogates is operational and therefore model-relative. Predictive uncertainty combines irreducible observation noise (including masking and measurement noise) with residual structure arising from approximation limits, optimization error, and data scarcity. Generalization uncertainty captures the sensitivity of the fitted model to train-specific patterns that fail to transfer to held-out data, resembling an epistemic component. Here it is learned by optimizing uncertainty controls with regularization-split gradients, without requiring a prior over weights or a posterior approximation. Appendix B discusses implications for chaotic regimes and clarifies the distinction from conventional hyperparameter optimization.

## 8 Conclusion

We presented cross-regularized uncertainty learning for neural PDE surrogates, where uncertainty parameters are trained online using regularization-split gradients while predictor parameters are trained on the training split. This train/reg routing yields an operational separation between predictive uncertainty (fit-time residual absorption) and generalization uncertainty (train–held-out mismatch control) without ensembles or Bayesian posterior approximations. Empirically, the resulting predictive distributions improve one-step Gaussian-mixture calibration under partial observability and limited data, and spatial diagnostics show uncertainty concentrating in high-error regions. Extending these guarantees from one-step teacher-forced evaluation to calibrated long-horizon autoregressive rollouts requires propagating uncertainty through compounding model error; we leave that setting to future work.

A related direction is richer predictive noise inside the model interior. Predictive noise injected at internal layers can, in principle, induce correlated or multimodal output uncertainty. Extending the current instantiations in that direction may narrow the gap between single-model uncertainty learning and richer uncertainty families used in scientific decision-making.

## References


Anastasios N. Angelopoulos and Stephen Bates. A gentle introduction to conformal prediction and distribution-free uncertainty quantification. *arXiv preprint arXiv:2107.07511*, 2021.

Carlos Stein Brito. Cross-regularization: Adaptive model complexity through validation gradients. *ICML*, 2025.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016.

Felix Koehler, Simon Niedermayr, Rüdiger Westermann, and Nils Thuerey. APEBench: A benchmark for autoregressive neural emulators of PDEs. *Advances in Neural Information Processing Systems*, 38, 2024.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NeurIPS*, 2017.

Xutong Li, Zongyi Li, Nikola Kovachki, and Anima Anandkumar. Geometric operator learning with optimal transport. *arXiv preprint arXiv:2507.20065*, 2025.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, et al. Fourier neural operator for parametric partial differential equations. *ICLR*, 2021.

Lu Lu, Pengzhan Jin, George Pang, et al. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 2021.

Ziqi Ma, David Pitt, Kamyar Azizzadenesheli, and Anima Anandkumar. Calibrated uncertainty quantification for operator learning via conformal prediction. *Transactions on Machine Learning Research*, 2024.

Albert Matveev, Sanmitra Ghosh, Aamal Hussain, James-Michael Leahy, and Michalis Michaelides. Light-weight diffusion multiplier and uncertainty quantification for fourier neural operators. *arXiv preprint arXiv:2508.00643*, 2025.

PhysicsX. Dinozaur dataset repository. `https://huggingface.co/datasets/PhysicsX/DINOZAUR`, 2025.

Makoto Takamoto et al. Pdebench: An extensive benchmark for scientific machine learning. In *NeurIPS Datasets and Benchmarks*, 2022.

# A Appendix: Datasets, Splits, and Reproducibility

## A.1 Default Training Configuration

Unless explicitly overridden in a sweep axis, runs use:

- a 4-layer FNO backbone with 12 Fourier modes, hidden width 8, and GELU activations,
- batch size 16,
- total optimization steps 30,000,
- evaluation interval 250 steps,
- one regularization update every 5 training steps,
- 10 Monte Carlo samples for train, regularization, and test evaluations,
- learning rates $10^{-3}$ for backbone weights, $10^{-2}$ for internal predictive-noise scales, $10^{-2}$ for internal generalization-noise scales, and $10^{-3}$ for head-level generalization-noise scales,
- initialization $\log \sigma_{\text{pred}} = -5$, $\log \sigma_{\text{gen}} = -5$.

## A.2 APEBench Data and Task Construction

All primary experiments are built from APEBench (Koehler et al., 2024) one-dimensional trajectory data under the `diff_ks` setting (160 spatial points). From each trajectory, we construct one-step supervised pairs $(x_t, x_{t+1})$ for teacher-forced training and evaluation. Unless stated otherwise, train horizon is 10 and test horizon is 200. We report one-step metrics directly on these pairs because the model is trained to quantify one-step uncertainty. Long-horizon rollout calibration is future work and is nontrivial: uncertainty must remain calibrated after repeated autoregressive propagation with compounding prediction error.

## A.3 OTNO Car Data Provenance

The car-pressure experiment uses the ShapeNetCar slice from the DINOZAUR data release (Matveev et al., 2025; PhysicsX, 2025), using the local `data/shapenetcar` split prepared for these experiments. In this setup, each sample is a car-surface point set with geometry/flow-derived inputs and scalar surface pressure targets. Our OTNO model follows the neuraloperator implementation protocol, and OTNO denotes *Optimal Transport Neural Operator* (Li et al., 2025).

## A.4 Train/Reg/Test Splits

The optimization protocol uses an explicit three-way split:

- **train split:** used to optimize deterministic model weights and predictive uncertainty parameters;
- **regularization set (reg. set):** used only for updates of generalization-noise parameters via regularization objectives;
- **test split:** held out for final one-step and calibration reporting.

This split is central to the method claim: predictive and generalization uncertainty are separated by data routing and objective routing, with parameter naming reflecting that separation.

### A.5 Observation Masking Protocol

Spatial masking is controlled by the observed fraction of input and target fields. We use a controlled-occlusion protocol: for each run, masks are sampled once with a fixed seed and reused throughout training to remove mask stochasticity from method comparisons. In all sweep plots, observation settings are matched between compared methods (XReg vs. MC dropout), so differences are attributable to uncertainty-learning strategy with data visibility held fixed. Random-per-step masking is a relevant extension for future work.

### A.6 Evaluation Metrics and Logging Conventions

Mixture calibration is measured with

$$\mathrm{ECE}_{\mathrm{mix}} = \frac{1}{|\mathcal{A}|} \sum_{\alpha \in \mathcal{A}} \left| \widehat{\mathrm{Coverage}}(\alpha) - \alpha \right|,$$

where $\mathcal{A} = \{0.1, \ldots, 0.9\}$. For each $\alpha$, $\widehat{\mathrm{Coverage}}(\alpha)$ is empirical central-interval coverage computed from the Gaussian-mixture predictive CDF and averaged over space-time one-step targets. One-step NLL is reported per spatial target point.

We report:

- one-step test NLL from MC predictive evaluation,
- mixture calibration error (ECE) on regularization and test splits,
- calibration curves over nominal coverage grid points,
- uncertainty decomposition statistics including $\mathrm{Std}_{\omega}[\mu]$ and layerwise generalization-noise scales.

### A.7 Computational Overhead

XReg adds a periodic regularization update for $\rho$. If one train update has cost $C_{\text{train}}$, one regularization update has cost $C_{\text{reg}}$, and regularization is applied every $k_{\text{reg}}$ steps, the approximate overhead factor is

$$\text{overhead} \approx 1 + \frac{1}{k_{\text{reg}}} \frac{C_{\text{reg}}}{C_{\text{train}}}.$$

In our default setting ($k_{\text{reg}} = 5$), this is a moderate overhead because $\rho$-updates are sparse and low-dimensional. The cost can be reduced further by increasing $k_{\text{reg}}$ (less frequent regularization updates). Prior empirical reports indicate that practical settings can approach roughly $1.1\times$ standard training cost (Brito, 2025), comparable to ordinary training pipelines that already include periodic validation checks.

## B Appendix: Extended Discussion on Interpretation and Optimization

This section expands two discussion points that are secondary to the main empirical claims.

**Model-relative uncertainty in deterministic regimes.** The model-relative interpretation is especially relevant in domains where "intrinsic noise" is debated. Chaotic systems are an illustrative case: the governing dynamics may be deterministic, yet finite data and finite-capacity models still produce large predictive residuals. In this setting, predictive uncertainty tracks unexplained residual structure, while generalization uncertainty tracks overfitting to the specific training sample.

**Why adaptation remains stable.** The adaptation mechanism learns uncertainty magnitude from observed train–held-out mismatch during optimization. In our setup, this remains stable because regularization variables are low-dimensional noise controls, not full feature maps; the regularization step adjusts uncertainty scales without re-fitting the full predictor on held-out data.

**Relation to hyperparameter optimization.** XReg performs direct parameter learning of uncertainty controls within a single training process. Classical hyperparameter tuning optimizes external

meta-parameters (e.g., a fixed $\lambda$ selected by outer-loop search), whereas XReg updates model parameters that control uncertainty. In Algorithm 1, $(\theta, \psi)$ are updated on train data, and $\rho$ is updated online from reg. set gradients.

## C Appendix: OTNO Car Uncertainty Diagnostics

This appendix reports optimization-time uncertainty-scale trajectories and final uncertainty magnitudes for the OTNO car experiment in Figure 5. The goal is to show that both predictive and generalization branches remain active and non-degenerate in the data-limited car regime.

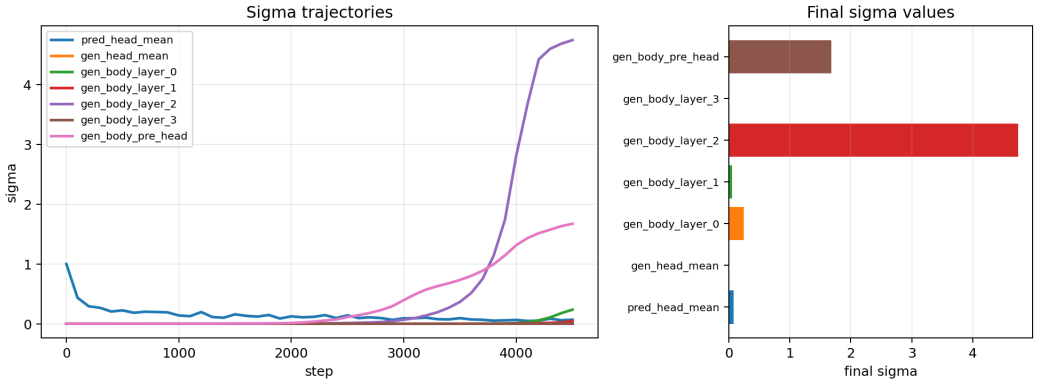



Figure 8: Sigma diagnostics for the car experiment shown in Figure 5. Left: trajectory of learned uncertainty scales during optimization. Right: final values across uncertainty components. The run exhibits non-degenerate predictive and generalization scales, with clear growth in selected generalization components and no collapse to near-zero values.

## D Appendix: Sweep Calibration Panels

This appendix provides full calibration montages for all runs in the XReg and MC-dropout sweeps summarized in the main text tables and aggregate figures. These panels serve as a full visual audit trail beyond scalar ECE summaries.

| Train Size | Method | Test NLL (MC)↓ | Test $\text{ECE}_{\text{mix}}$ ↓ | Reg $\text{ECE}_{\text{mix}}$ ↓ |
|---|---|---|---|---|
| 20 | XReg | -0.3277 | **0.0965** | **0.0867** |
| 20 | MC Dropout | **-0.4343** | 0.1254 | 0.1197 |
| 20 | Deep Ensemble (3) | 0.7512 | 0.1344 | 0.1361 |
| 30 | XReg | -0.5962 | **0.0273** | **0.0210** |
| 30 | MC Dropout | **-0.6894** | 0.0857 | 0.0815 |
| 30 | Deep Ensemble (3) | 0.1574 | 0.1157 | 0.1131 |
| 40 | XReg | -0.6671 | **0.0144** | **0.0118** |
| 40 | MC Dropout | **-0.8858** | 0.0375 | 0.0410 |
| 40 | Deep Ensemble (3) | -0.7355 | 0.0760 | 0.0727 |
| 70 | XReg | -1.4173 | **0.0118** | **0.0128** |
| 70 | MC Dropout | -1.1361 | 0.0666 | 0.0654 |
| 70 | Deep Ensemble (3) | **-2.0064** | 0.0569 | 0.0555 |

Table 2: Numeric comparison for the train-size sweep at fixed observed fraction $0.7$. XReg maintains substantially lower test/reg mixture ECE across all four train sizes, including regimes where baseline one-step NLL is competitive.

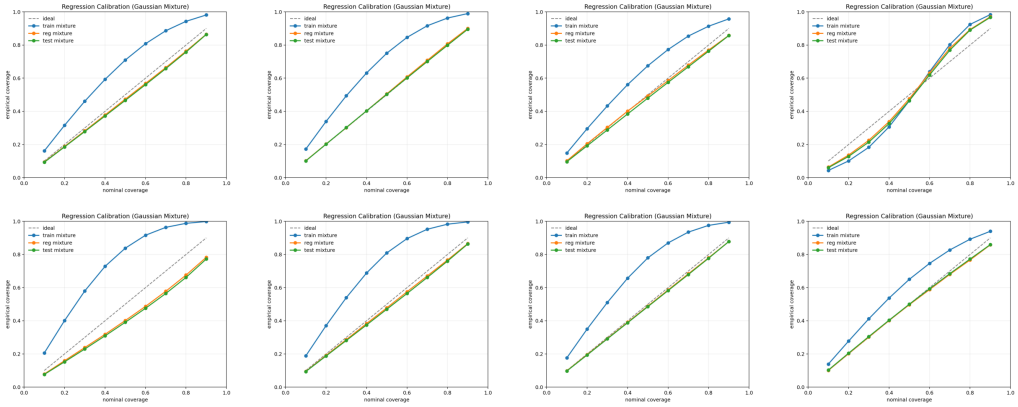



Figure 9: Full XReg sweep calibration montage (regression-mixture calibration for all eight XReg runs in the 10k-step sweep reported in the main text). These panels are included as a raw visual audit trail: they expose per-run curve shape differences that are not fully captured by scalar ECE summaries.

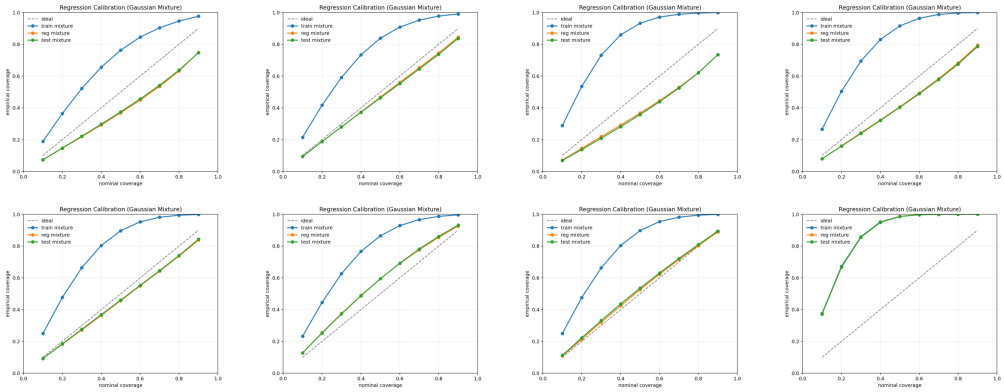



Figure 10: MC-dropout sweep calibration montage for the full completed 8-run sweep (observed-fraction axis and train-size axis, dropout $p = 0.1$). Along with Figure 9, these panels provide a many-panel calibration comparison complementing the aggregate loss/ECE table in the main text.

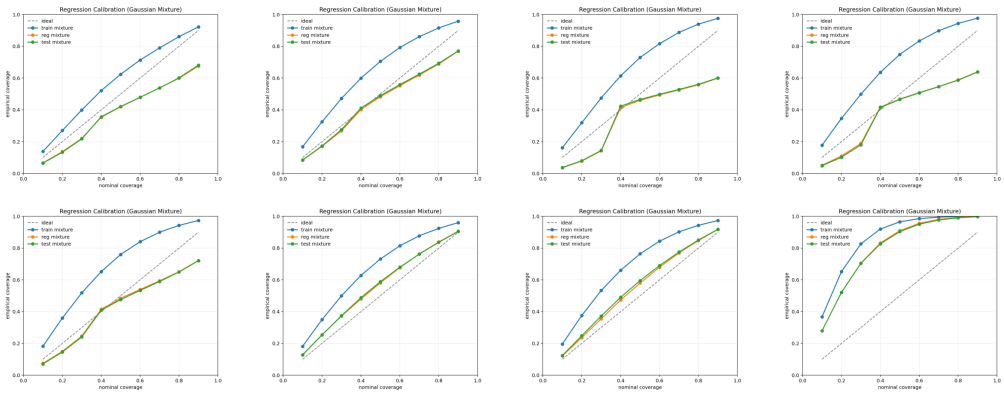



Figure 11: Deep-ensemble sweep calibration montage for the full completed 8-point sweep (observed-fraction axis and train-size axis), with each point evaluated as a 3-member aggregated predictive mixture. This view complements Figures 9 and 10 with a full-panel visual audit for the ensemble baseline.