DIRECTLY OPTIMIZING CALIBRATED TEST-TIME UNCERTAINTY

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

ABSTRACT

Uncertainty in learned predictors is often split into aleatoric and epistemic components using architectural choices or Bayesian approximations, making the decomposition sensitive to modeling details. We propose an objective-driven decomposition into *predictive noise* (ψ) and *generalization noise* (ϕ). Predictive noise represents the residual stochasticity needed to fit the training data under a chosen likelihood and model class, while generalization noise captures instability of the learned predictor as revealed by held-out data. Both noises can be instantiated as additive randomness in the predictive distribution (output-only or internal), and they remain separable because they are assigned different optimization roles: predictive noise is learned as part of the training likelihood for (θ, ψ), while generalization noise is learned from a held-out marginal log-likelihood in the spirit of cross-regularization. The regularization split serves as a proxy for the unseen conditions under which uncertainty will be consumed, so optimizing ϕ on that split directly calibrates the test-time predictive distribution. In our experiments, the resulting total predictive distribution yields improved reliability in a single-model setting and produces *noise-budget curves* that help interpret how performance changes across data size and capacity. We instantiate the framework in a controlled mixture model and in regression, and include a structured-perturbation case study in segmentation.

1 INTRODUCTION

Reliable systems require accurate point predictions together with predictive distributions that reflect uncertainty. Yet, in practice, uncertainty is frequently decomposed in ways that are *implementation-defined*. A common convention associates "aleatoric" uncertainty with explicit output noise and "epistemic" uncertainty with parameter randomness approximated via Bayesian surrogates or explicit ensembles (Kendall & Gal, 2017; Gal & Ghahramani, 2016; Lakshminarayanan et al., 2017). This convention makes the split sensitive to modeling choices (e.g., where randomness is injected) and can obscure what each component is meant to explain.

We introduce two learnable noises defined operationally: **predictive noise** ψ and **generalization noise** ϕ. Predictive noise is the stochasticity needed to fit the training distribution under the chosen likelihood and model class; it absorbs ambiguity and misspecification *relative to the learned predictor*. Generalization noise captures predictor instability as revealed by held-out data and is learned to avoid overconfident failures. At test time we use the **total predictive distribution**, with both ψ and ϕ active. The split is carried by objective role across output-only, internal, and co-located instantiations, so distinct objectives assign the two noises distinct explanatory roles. The regularization split serves as a proxy for unseen conditions, and optimizing ϕ on that split is designed to improve calibration of the predictive mixture that will ultimately be used at test time. Tracking the resulting noise budgets across data size and capacity yields learning curves that explain *why* held-out performance changes regime. We give an objective-based decomposition into predictive and generalization noise. We also give a single-run training procedure that learns (θ, ψ) on train NLL and ϕ on held-out marginal log-likelihood, together with *noise-budget curves* that interpret performance trends across data size and capacity.

2 OBJECTIVE-BASED NOISE DECOMPOSITION

Two-noise predictive model. Let a predictor with parameters θ map inputs to distribution parameters $\eta_\theta(x)$ (logits for classification; mean/scale for regression). Introduce two independent noise seeds: $\epsilon_\psi \sim \mathcal{N}(0, I)$ for predictive noise and $\epsilon_\phi \sim \mathcal{N}(0, I)$ for generalization noise. In general, these noises may act on weights, hidden representations, or output parameters through a stochastic predictive-parameter map $\tilde{\eta}(x; \theta, \psi, \phi, \epsilon_\psi, \epsilon_\phi)$. A simple additive output-parameter instantiation is

$$\eta(x) = \eta_\theta(x) + s_\psi(x) \odot \epsilon_\psi + s_\phi(x) \odot \epsilon_\phi, \tag{1}$$

where $s_\psi(x)$ and $s_\phi(x)$ are learned noise magnitudes and $\odot$ is elementwise multiplication. Given a base likelihood family $p_0(y \mid \eta)$ (categorical, Gaussian, MDN, ...), define the **total predictive distribution** by marginalizing both noises:

$$p(y \mid x; \theta, \psi, \phi) = \mathbb{E}_{\epsilon_\psi, \epsilon_\phi}\left[p_0\big(y \mid \tilde{\eta}(x; \theta, \psi, \phi, \epsilon_\psi, \epsilon_\phi)\big)\right]. \tag{2}$$

Example: mixture-of-lines with co-located noises. In the controlled curve-fitting setting used in Figure 1, the predictor is a mixture of two affine components. In this example, we write predictive-noise draws as ξ and generalization-noise draws as ϵ. For component $z \in \{0, 1\}$, one convenient additive instantiation is

$$\mu_z(x) = \big(w_z + \sigma_{\text{pred}}^{w_z}\xi_z^w + \sigma_{\text{gen}}^{w_z}\epsilon_z^w\big)x + \big(b_z + \sigma_{\text{pred}}^{b_z}\xi_z^b + \sigma_{\text{gen}}^{b_z}\epsilon_z^b\big) + \sigma_{\text{gen}}^{y_z}\epsilon_z^y, \tag{3}$$

with an observation model

$$y \mid x, z, \xi, \epsilon \sim \mathcal{N}\big(\mu_z(x), (\sigma_{\text{pred}}^{y_z})^2\big), \qquad z \sim \text{Bernoulli}(\pi). \tag{4}$$

Here predictive and generalization noises are co-located on the same slope/intercept terms; the distinction comes from which objective trains them.

Operational meaning. **Predictive noise** ψ is trained to explain *training-fit residuals*: ambiguity, likelihood misspecification, and residual unpredictability relative to the model class. **Generalization noise** ϕ is trained to explain *held-out overconfidence and misfit*: it increases uncertainty where the learned predictor is unstable with respect to the data. Because ψ and ϕ can be co-located and share the same functional form, the split is carried by their objectives within a shared architecture.

Distinct objectives make the two noises learnable in the population setting. With finite data, the split also depends on how distinctly the two perturbations affect the predictive mixture, so mild budget tradeoffs or oscillations can appear before the decomposition stabilizes.

Noise budgets. We summarize each noise by a scalar budget,

$$B_\psi = \mathbb{E}_{x \sim \mathcal{D}}\left[\|s_\psi(x)\|_1\right], \qquad B_\phi = \mathbb{E}_{x \sim \mathcal{D}}\left[\|s_\phi(x)\|_1\right], \tag{5}$$

and study how (B_ψ, B_ϕ) evolve with data size and model capacity alongside held-out NLL. These **noise-budget learning curves** provide a mechanistic account of learning beyond a single performance curve.

3 TRAINING METHOD

We learn a single model by splitting optimization roles across data splits. Let $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{val}}$ be a train/held-out split.

Training objective for (θ, ψ). We train (θ, ψ) to fit the training distribution under a proper scoring rule (negative log-likelihood). To keep notation compact, define the *predictive likelihood*

$$p_\psi(y \mid x, \epsilon_\phi; \theta, \psi, \phi) = \mathbb{E}_{\epsilon_\psi}\left[p_0\big(y \mid \tilde{\eta}(x; \theta, \psi, \phi, \epsilon_\psi, \epsilon_\phi)\big)\right]. \tag{6}$$

This is the predictive distribution after integrating predictive noise for a fixed realization of generalization noise, regardless of whether the noises are injected in weights, hidden units, or output parameters. The training objective is

$$\mathcal{L}_{\text{train}}(\theta, \psi; \phi) = \mathbb{E}_{(x,y) \sim \mathcal{D}_{\text{train}}}\, \mathbb{E}_{\epsilon_\phi}\left[-\log p_\psi(y \mid x, \epsilon_\phi; \theta, \psi, \phi)\right]. \tag{7}$$

In words, predictive noise is the part of the model used to absorb training-fit residuals under the chosen likelihood, while generalization noise acts as a complexity perturbation whose effect is averaged over training examples.

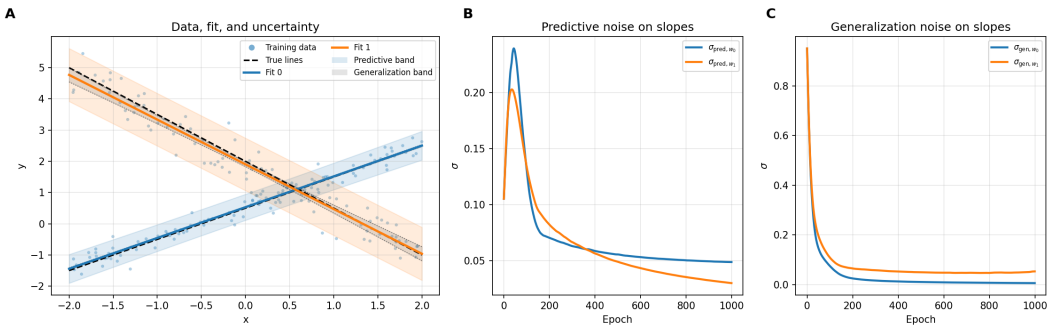


Figure 1: Mixture model results under the correct objective pairing. Predictive and generalization noise separate and jointly form the total predictive distribution.

Held-out marginal objective for ϕ. Generalization noise ϕ is optimized on held-out data using the marginal log-likelihood of the same predictive family:

$$\mathcal{L}_{\text{val}}(\phi; \theta, \psi) = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{val}}} \left[-\log \mathbb{E}_{\epsilon_\phi} \left[p_\psi(y \mid x, \epsilon_\phi; \theta, \psi, \phi) \right] \right]. \tag{8}$$

With K samples $\{\epsilon_\phi^{(k)}\}_{k=1}^K$, we estimate this objective with log-mean-exp:

$$\mathcal{L}_{\text{val}}(\phi; \theta, \psi) \approx \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{val}}} \left[-\log \left(\frac{1}{K} \sum_{k=1}^{K} p_\psi(y \mid x, \epsilon_\phi^{(k)}; \theta, \psi, \phi) \right) \right]. \tag{9}$$

This objective is the log-likelihood of the predictive mixture induced by generalization noise and encourages ϕ to expand uncertainty where the learned predictor is unstable.

Held-out tuning of ϕ. The held-out objective scores the same marginal predictive distribution used at test time, so ϕ is enlarged where it improves held-out log score and remains small where the learned mapping is already well supported (Gneiting & Raftery, 2007; Guo et al., 2017; Stein Brito, 2025). In this sense, the regularization split acts as a proxy for the unseen conditions under which uncertainty will be consumed, and the update on ϕ is optimized to improve calibration of the final test-time predictive mixture during training. This also gives ϕ a natural ensemble-like interpretation: it acts as a single-run proxy for the predictive variability that repeated retraining or an explicit ensemble would reveal, learned directly through held-out marginal likelihood. In practice we alternate updates: (θ, ψ) are optimized on $\mathcal{L}_{\text{train}}$ with batches from $\mathcal{D}_{\text{train}}$, then ϕ is optimized on $\mathcal{L}_{\text{val}}$ with batches from $\mathcal{D}_{\text{val}}$. Small K already suffices for stable held-out log-mean-exp estimation in our experiments.

4 Experiments

Mixture model. We use a controlled mixture setting where model capacity and data size can be varied cleanly. Figure 1 shows learned predictive and generalization noise behavior under the correct objective pairing, together with the resulting predictive distribution. We report held-out NLL and noise budgets B_ψ, B_ϕ (with calibration analysis in the appendix). The learned budgets separate cleanly: B_ψ tracks residual fit while B_ϕ reflects held-out instability, and their evolution explains changes in held-out NLL across regimes. A swap ablation that exchanges the objectives degrades held-out behavior (Appendix Table 1).

Calibration metric. We report Expected Calibration Error (ECE; equal-width bins) and reliability diagrams on CTR23, a curated tabular regression benchmark suite; here we use the OpenML diamonds task 361257 (Appendix). Calibration is one of the main practical payoffs of the framework: the regularization split serves as a proxy for test-time conditions, and the held-out objective tunes ϕ on the same marginal predictive distribution that is used at test time. On CTR23, the held-out and test reliability curves lie close to the diagonal (Appendix Figure 9), producing a better-calibrated test-time predictor in our experiments from a single model through the training objective itself.

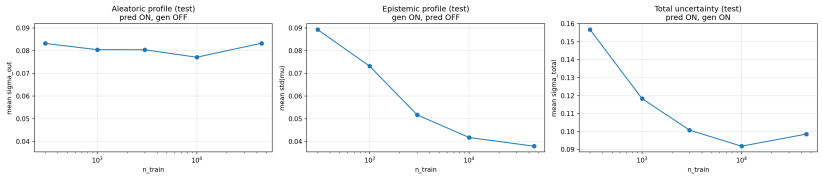


Figure 2: Noise-budget curves as a function of training-set size on a tabular regression benchmark. Generalization noise decreases with more data, while predictive noise remains comparatively stable, indicating that the two losses learn distinct sources of uncertainty.

Noise-budget curves. Figure 2 shows predictive and generalization noise behavior on CTR23 with a fixed 15% test split and varying n_{train} at fixed capacity. Only generalization noise decreases strongly with data size, while predictive noise remains comparatively stable. This is the signature of the split objectives: limited-data instability is carried by ϕ, whereas ψ captures a residual, model-relative uncertainty floor. The total uncertainty follows the contraction of ϕ and approaches the predictive floor as the training set grows.

This pattern matches the appendix sanity checks. In a constant model, generalization noise follows the expected $1/\sqrt{N}$ contraction while predictive noise stays near the data-noise floor (Appendix Figures 3 and 4). In a stochastic-slope toy, irreducible parameter variability is allocated to predictive noise, again with generalization noise contracting as data accumulate (Appendix Figure 6).

Structured predictive variability. The same split is useful beyond scalar output uncertainty. In low-data MLP regression, predictive noise helps represent the modes supported by the training distribution while generalization noise broadens uncertainty into weak-support regions (Appendix Figure 7). In structured prediction, internal predictive noise supports coherent alternatives across the output field: the CamVid example in Appendix Figure 8 shows spatially consistent predictive variations under internal noise, and co-located generalization noise retains its role as held-out-tuned stability control.

5 Conclusion

We proposed an objective-based decomposition of uncertainty into predictive noise ψ and generalization noise ϕ. The components can be co-located, and their separability arises from training ψ as the uncertainty needed to explain training-fit residuals and training ϕ through a held-out marginal objective that scores predictive reliability. Because the regularization split serves as a proxy for unseen conditions, optimizing ϕ there is designed to improve calibration of the predictive mixture used at test time. This yields noise-budget learning curves that explain performance trends across data size and capacity, and improves predictive reliability in a single-model setting while remaining flexible about where and how uncertainty is injected.

References

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, 2016.

Tilmann Gneiting and Adrian E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.

Alex Kendall and Yarin Gal. What uncertainties do we need in Bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Carlos Stein Brito. Cross-regularization: Adaptive model complexity through validation gradients. In *International Conference on Machine Learning (ICML)*, 2025.

A Additional derivations and intuition

A.1 Train likelihood and held-out marginal likelihood

To keep the notation compact, define

$$p_\psi(y \mid x, \epsilon_\phi; \theta, \psi, \phi) = \mathbb{E}_{\epsilon_\psi}\left[p_0\big(y \mid \tilde{\eta}(x; \theta, \psi, \phi, \epsilon_\psi, \epsilon_\phi)\big)\right]. \tag{10}$$

Here $\tilde{\eta}$ denotes the noisy predictive parameters induced by the chosen injection scheme, which may act on weights, hidden states, or output parameters. This is the predictive likelihood after integrating predictive noise for a fixed realization of generalization noise. The key technical distinction is then:

$$\mathcal{L}_{\text{train}}(\theta, \psi; \phi) = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{train}}}\, \mathbb{E}_{\epsilon_\phi}\left[-\log p_\psi(y \mid x, \epsilon_\phi; \theta, \psi, \phi)\right], \tag{11}$$

while generalization noise ϕ is optimized with the held-out marginal objective

$$\mathcal{L}_{\text{val}}(\phi; \theta, \psi) = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{val}}}\left[-\log \mathbb{E}_{\epsilon_\phi}\left[p_\psi(y \mid x, \epsilon_\phi; \theta, \psi, \phi)\right]\right]. \tag{12}$$

The train objective asks for a predictor that fits each held-fixed realization of generalization noise well on average over the training split. The held-out objective scores the full marginal predictive distribution induced by generalization noise. This "inside-the-log" objective can be estimated via log-mean-exp with K samples (Section 3).

A.2 Why the marginal objective calibrates test-time uncertainty

Consider a point (x, y) where the deterministic predictor is mis-specified, so $p_0(y \mid \eta_\theta(x))$ is extremely small. Under the held-out marginal objective, the model is rewarded for assigning mass through the *predictive mixture*. Increasing ϕ broadens that mixture where the learned predictor is unstable, which improves support on unseen points and reduces overconfident errors under proper scoring rules (Gneiting & Raftery, 2007; Guo et al., 2017).

A complementary view uses Jensen's inequality:

$$-\log \mathbb{E}[p] \le \mathbb{E}[-\log p]. \tag{13}$$

The held-out objective therefore evaluates the log score of the *marginal predictive distribution itself*. This is exactly the object of interest at test time when both noises are active. The regularization split therefore plays the role of a proxy for the unseen conditions under which this predictive mixture will be used, and optimizing ϕ on that split directly calibrates test-time uncertainty.

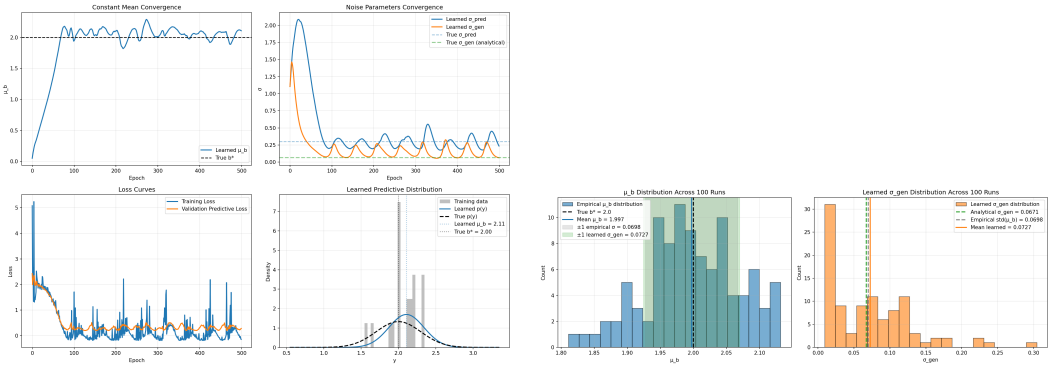
Distinct objectives make the two noises learnable in the population setting. With finite data, the learned split also depends on how distinctly predictive and generalization perturbations affect the predictive mixture. When those effects are close in scale, part of the uncertainty budget can move between the two components before stabilizing, which is consistent with the small oscillations visible in some appendix plots.

A.3 Connection to ensembles and retraining variability

Deep ensembles approximate uncertainty by training multiple predictors and averaging their predictive densities (Lakshminarayanan et al., 2017). In our formulation, ϕ acts as a learnable single-run proxy for the predictive variability induced by repeated retraining. When ϕ is learned by held-out marginal likelihood, the model is incentivized to match the mixture behavior that an explicit ensemble would provide, while keeping compute and engineering minimal.

Setup	Test NLL (mean $\pm$ std)	Notes
Correct pairing	0.7864 ± 0.0189	20 runs, $n_{\text{test}} = 5000$
Swapped pairing	0.8101 ± 0.0765	higher error + variance

Table 1: Mixture-of-lines swap ablation. Swapping the loss assignments yields worse predictive NLL, confirming that the objective split carries the semantic role of each noise even when the two are co-located.



(a) Single run fit

(b) Recovery across runs

Figure 3: Constant toy sanity check: predictive noise fits the noise floor while generalization noise shrinks with data.

B Conceptual framing and modeling choices

We use the terms *predictive noise* (ψ), *generalization noise* (ϕ), and *total predictive uncertainty* to keep the decomposition operational. Predictive noise is variability in outputs produced by a fixed learned predictor. Generalization noise is variability in the learned predictor due to finite data and optimization instability, exposed through held-out behavior. Total predictive uncertainty is the uncertainty of the predictive distribution that downstream users consume when both effects are active.

This framing separates meaning from *where* randomness is injected. Output-only noise is often sufficient for simple targets; structured outputs or multi-modal regression may benefit from internal stochasticity or latent variables that represent predictive variability coherently. Generalization noise can be represented by weight-level randomness, representation-level noise, or lower-dimensional proxies, as long as it affects predictions in the right way. The key separation is objective-driven: ψ is trained to explain training-fit residuals, while ϕ is trained through a held-out proper-scoring objective that measures predictive reliability.

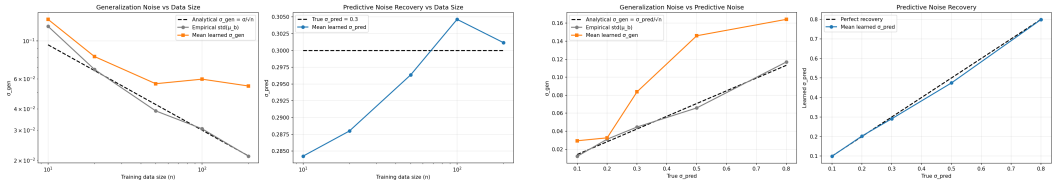
C Additional controlled experiments

C.1 Objective pairing ablation

We evaluate swapping which objective learns ψ versus ϕ. Exchanging the roles breaks the operational meaning of the noises and degrades test NLL.

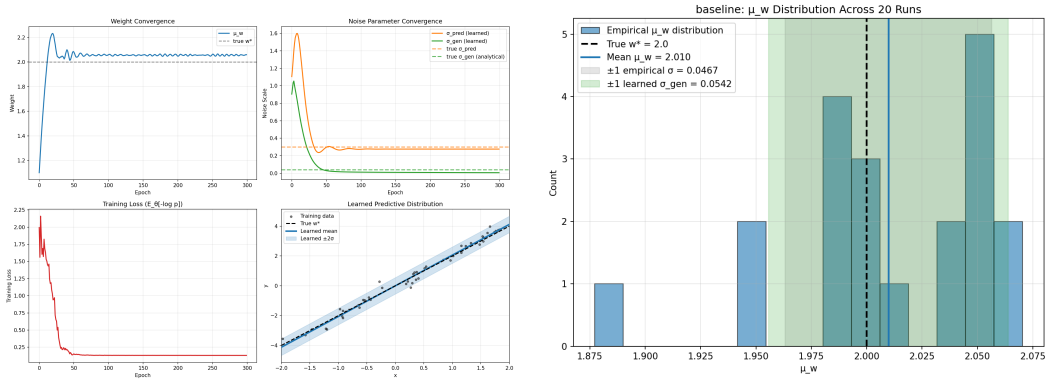
C.2 Constant model: 1/sqrt(N) scaling

We train a constant predictor $y = \mu + \sigma_{\text{pred}}\,\xi + \sigma_{\text{gen}}\,\epsilon$ on synthetic data and vary N. Predictive noise σ_{pred} matches the data noise floor, while generalization noise σ_{gen} shrinks with N and empirically follows the standard-error scaling $1/\sqrt{N}$. Figure 3 shows a single run and recovery across runs; Figure 4 shows the data-size and predictive-noise sweeps.



(a) Data-size sweep (b) Predictive-noise sweep

Figure 4: Scaling behavior for the constant model. Generalization noise contracts with data size while predictive noise stabilizes at the data noise floor.



(a) Single run fit (b) Recovery across runs

Figure 5: Linear regression sanity check with co-located predictive and generalization noise.

C.3 Linear regression sanity check

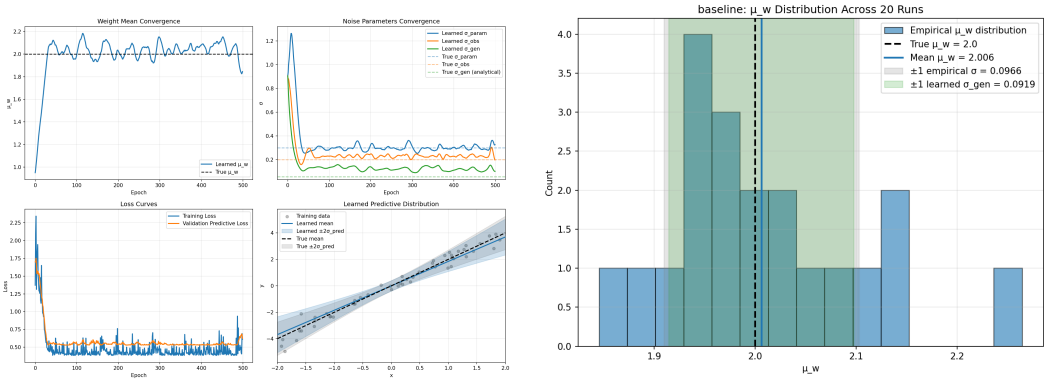
We repeat the decomposition in linear regression, where the predictive and generalization components are co-located. The objective split still separates the roles: predictive noise explains residual fit; generalization noise expands uncertainty to cover held-out discrepancies. Figure 5 shows the baseline fit and recovery.

C.4 Stochastic slope: intrinsic parameter variability

We generate data from a linear process with a truly stochastic slope, so the slope variance is irreducible. The decomposition correctly allocates this variability to predictive noise on the slope, while generalization noise vanishes as data increase. Figure 6 shows the baseline run and recovery across runs.

D MLP regression: predictive noise on/off

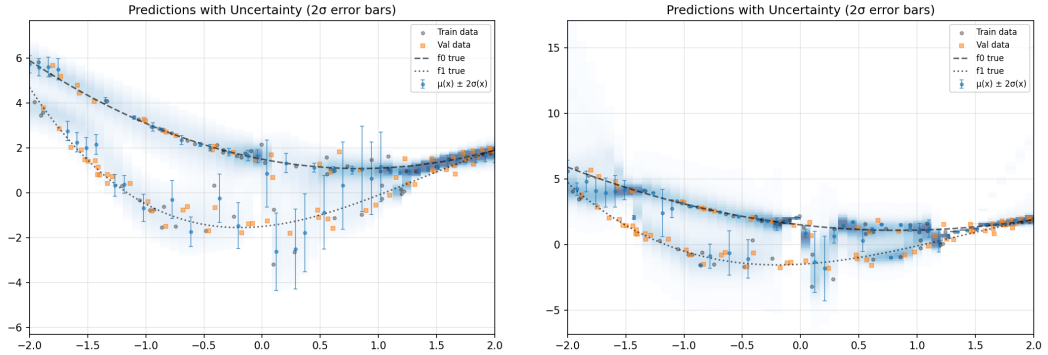
We visualize a low-data MLP regression setting with generalization noise active and compare predictive noise on vs. off. This example makes clear that the two noises do different jobs even within the same predictive family. Predictive noise covers the modes supported by the training data, including the multi-modal mixture itself, while generalization noise expands coverage into regions with little or no training support and improves held-out reliability. The contrast is useful for reading the structured-noise example below: predictive noise serves as a mechanism for representing plausible alternatives supported by the data.



(a) Single run fit

(b) Recovery across runs

Figure 6: Stochastic slope toy: intrinsic parameter variability is captured by predictive noise, while generalization noise contracts with data.



(a) Predictive noise on

(b) Predictive noise off

Figure 7: Low-data MLP regression with generalization noise active. Error bars show $\mu(x) \pm 2\sigma(x)$.

E Structured noise with delta samples

We include an example of structured predictive noise using delta-style sampling. The point of this example is breadth across tasks together with a clear illustration of why internal predictive noise can matter. For structured outputs such as segmentation, output-only uncertainty scales mainly widen local marginals. Internal predictive noise can induce correlated changes across pixels and object boundaries, yielding plausible alternative labelings with spatial coherence. CamVid (semantic segmentation benchmark) provides a simple illustration: Figure 8 shows one example (index 100) where internal predictive perturbations produce spatially coherent variations. This is exactly the setting where co-located predictive and generalization noises are useful: they may act in the same internal pathway, and their roles remain distinct because predictive noise is optimized to explain training-supported ambiguity and generalization noise is optimized to improve held-out reliability.

F Implementation details

F.1 Noise parameterizations

We use additive noise on distribution parameters for simplicity, but the framework supports:

- **Output-only noise** on logits or mean/scale (often sufficient for single-output tasks).
- **Internal injection** on intermediate representations for structured predictive variability.

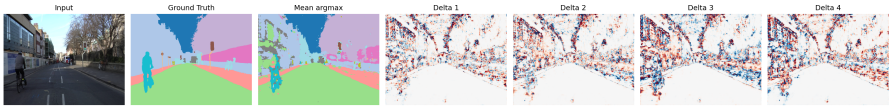


Figure 8: Structured delta samples illustrating coherent predictive variability (CamVid).

- **Co-location** (same injection site) to show objective-level separability within a shared pathway.
- **Separate injection sites** (optional) when inductive bias suggests it.

Input-conditioning can be implemented as small heads producing $s_\psi(x)$ and $s_\phi(x)$.

F.2 Monte Carlo estimation

For training, we use a modest MC budget to integrate predictive noise while keeping the generalization perturbation fixed within each estimator. For held-out marginal estimation, we use a larger MC budget to stabilize log-mean-exp; exact sample counts are experiment-specific. We compute log-mean-exp stably via

$$\log\left(\frac{1}{K}\sum_k p_k\right) = m + \log\left(\frac{1}{K}\sum_k \exp(\log p_k - m)\right), \quad m = \max_k \log p_k. \tag{14}$$

F.3 Calibration metrics

We report NLL and at least one calibration metric. For classification, we use expected calibration error (ECE; equal-width bins) and reliability diagrams (Guo et al., 2017). For regression, we report interval coverage and calibration curves (empirical coverage vs. nominal) alongside sharpness.

Calibration is a primary empirical target of the framework and one of its main practical payoffs. The held-out objective optimizes the marginal predictive distribution that will actually be used at test time, so the method learns test-time calibration directly during training. Figure 9 illustrates this directly on CTR23: the held-out and test curves track the diagonal closely, indicating better-calibrated predictive mixtures at evaluation in our experiments. This behavior comes from the same single-model optimization loop, with the regularization split serving as a proxy for unseen conditions and guiding ϕ toward improved calibration of test-time uncertainty.

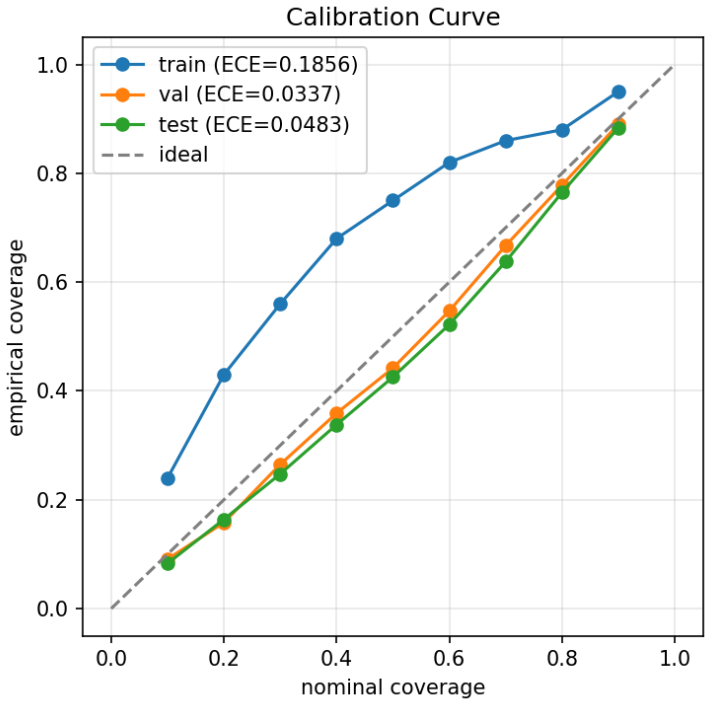


Figure 9: Reliability diagram on the tabular regression benchmark (OpenML diamonds), using equal-width bins. The held-out and test predictive mixtures remain close to the diagonal, illustrating calibrated test-time uncertainty learned directly by the training objective.