UNFREEZING THE PRIOR: ADAPTIVE PRIOR-DATA FITTED NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Prior-Data Fitted Networks (PFNs) learn to approximate Bayesian prediction from synthetic tasks generated under a training prior, but return only the final predictive distribution, not the posterior over the simulator choices that produced it. We call the ordered set of these choices a *semantic path*. Making the semantic path explicit allows us to replace a pretrained PFN's training prior when predicting for a new population of tasks. We introduce Adaptive-PFN, which separates inference over semantic paths from prediction conditional on them without practically losing training-prior predictive accuracy to a classical PFN while enabling post-training adaptation. Given a population of deployment data histories, we estimate a low-dimensional exponential tilt of the training prior and reweight the inferred distribution over semantic paths without updating network parameters. Adaptation without model checking, however, leaves a critical statistical question unanswered: does the assumed shift family actually explain the deployment population? We address this with a goodness-of-fit test. In controlled experiments with a complex synthetic prior, adaptation reduces mean squared error by 6.22% relative to a direct PFN with approximately the same number of parameters; the test empirically maintains nominal Type-I error and shows high power against prespecified departures from the adaptation model. In synthetic-to-real electricity forecasting, Adaptive-PFN models trained only on simulated data reduce active-hour weighted quantile loss (WQL) by 31.4–45.3% relative to a standard frozen PFN, without retraining. In retail forecasting, Adaptive-PFN also uses a deployment prior estimated independently from population-fitting stores and reduces WQL on open promotion days by 7.5% relative to the direct PFN. Further ablations provide evidence that the proposed method provides benefits beyond simpler output-level calibrations.

1 INTRODUCTION

A Prior-Data Fitted Network (PFN) (Müller et al., 2022) learns to approximate Bayesian prediction from simulated regression tasks drawn from a specified prior. Once trained, it can map a new data history $D = \{x_t, y_t\}_{t=1}^{N}$ and query x to a predictive distribution over the target y. This allows the PFN to produce predictions in a single forward pass without running Bayesian inference from scratch for each new task. This is useful in settings where running this kind of inference separately for each task is costly, like in time-series forecasting (Dooley et al., 2023) or tabular prediction (Hollmann et al., 2023; 2025).

In *deployment*, the trained model is applied to a new population of tasks which may differ from those sampled from the training prior. This can lead to suboptimal or poorly calibrated predictions. Past work has addressed temporal shifts through PFN pretraining (Helli et al., 2024), test-time correction for changing class frequencies (Lee, 2026), or inference conditioned on prior information supplied at test time, either explicitly (Whittle et al., 2026) or through related-task datasets sharing a common latent prior (Zhu et al., 2026). Appendix A relates Adaptive-PFN to these methods, empirical-Bayes population-prior estimation, and compatibility testing.

The training simulator for (D, x, y) generally contains a hierarchy of latent variables. A direct PFN is not exposed to these variables during training, and the resulting network does not explicitly represent their posterior distributions. We argue that this limits the ability of the learned network to adapt to certain shifts in deployment, which may be captured by a prior shift in a subset of latent variables.

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.

We call a user-selected subset of these variables the *semantic path*, $M = (M_1, \ldots, M_L)$. M is chosen to contain interpretable, typically high-level simulator choices whose population frequencies may plausibly differ at deployment and whose values can remain uncertain given a deployment dataset. For example, consider a linear-regression simulator with $y = Bax + \varepsilon$, where $B \in \{-1, +1\}$ is a slope sign, $a > 0$ its magnitude, and ε has a task-specific noise scale σ. We may take $M = (B, a)$, while leaving σ as an unexposed latent variable.

This paper introduces *Adaptive-PFN*. It separates inference of the semantic path from prediction conditional on it. The *semantic posterior network*, T_1, returns a distribution over semantic paths after observing D. The *conditional predictive network*, T_2, returns a predictive distribution over y at a query input x, conditional on D and a specified semantic path M. Averaging the predictions from T_2 using the distribution returned by T_1 approximates the Bayesian predictive distribution under the training prior (Section 2.1). The two networks have independent objectives and can be trained in parallel.

This separation provides three capabilities:

- **Infer semantic paths.** Adaptive-PFN returns the inferred distribution over semantic paths for each data history, rather than only its final predictive average.
- **Adapt without retraining.** A sample of deployment data histories is used to estimate a low-dimensional exponential tilt of the training prior over semantic paths (Section 2.2). Reweighting the output of T_1 by this tilt and averaging the unchanged predictions from T_2 adapts the pre-trained model without updating network weights (Section 2.3). This can be remarkably effective if the deployment shift is captured by a tilt in part of the semantic path; for instance, in the linear regression example above, if deployment tasks are more likely than training tasks to have a positive slope, $B = 1$. Alternatively, a deployment prior may be estimated via a domain-specific analysis and supplied directly as an input to this general reweighting procedure. This enables the application of a rich literature in empirical Bayes and hierarchical inference for particular model instances (Vandegar et al., 2021; Wagner-Carena et al., 2021; Teh et al., 2025; Cannella et al., 2026).
- **Check the prior-shift model.** The separation of T_1 and T_2 lets us test whether an exponential tilt in semantic-path frequencies is compatible with observed deployment data histories. We propose a goodness-of-fit test, which compares the observed and simulated deployment data histories through a prespecified representation (Section 2.5).

We evaluate these capabilities in a controlled experiment and two synthetic-to-real forecasting applications:

Controlled prior shifts. We first evaluate Adaptive-PFN in a controlled hierarchical model with 527,898 discrete semantic paths and one continuous semantic-path coordinate. Under the training prior, Adaptive-PFN and a direct PFN with approximately the same number of parameters have similar mean squared error (MSE): 2.1123 and 2.1160. After a prior shift, adaptation reduces MSE by 6.22% relative to the direct PFN.

Synthetic-to-real forecasting. We train the models only on synthetic time-series tasks and deploy them in two real domains. On Low Carbon London (Schofield et al., 2016), Adaptive-PFN reduces active-hour weighted quantile loss (WQL; Appendix F.6) by 31.4–45.3% across three seeds relative to the direct PFN. As a baseline, we compare against a calibrated direct PFN, which uses the same deployment data to rescale its predicted distribution. Adaptive-PFN reduces WQL by 12.5–38.2% relative to this baseline. On Rossmann Store Sales (Cukierski, 2015), we evaluate an Adaptive-PFN which estimates a deployment prior on semantic paths via a simple model-specific analysis. In this case, Adaptive-PFN reduces WQL on open promotion days by 7.48% across 256 stores relative to the direct PFN and lowers negative log-likelihood.

2 The Adaptive-PFN

Adaptive-PFN makes the semantic mixture inside Bayesian prediction explicit. Figure 1 summarizes its two-network architecture and adaptation at deployment time.

2.1 EXPOSING THE SEMANTIC MIXTURE

Throughout, the query input x is fixed and exogenous, and all probability statements are conditional on it.

For one task, let $D = \{(x_i, y_i)\}_{i=1}^{N}$ denote the observed data history. Let M denote the semantic path, with training prior $p_0(dm)$, and let $P_0(D, y \mid m, x)$ denote the simulator law of the data history and target conditional on $M = m$ at the fixed query x. These define a marginal prior over training data

$$P_0(D, y \mid x) = \int P_0(D, y \mid m, x)\, p_0(dm).$$

A direct PFN is a conditional predictive distribution $\pi_\theta(\cdot \mid x, D)$, with parameters θ, trained on tasks drawn from P_0. Let S be a strictly proper scoring rule, written in loss orientation. PFN training minimizes a Monte Carlo approximation to the population risk

$$\mathcal{L}_{\mathrm{PFN}}(\theta) = \mathbb{E}_{P_0}[S\{\pi_\theta(\cdot \mid x, D), y\}].$$

For the logarithmic score, $S(q, y) = -\log q(y)$. By strict propriety, the minimizer over all conditional distributions is the posterior predictive distribution under the training law, $P_0(dy \mid x, D)$. Thus, PFN training amortizes the Bayesian predictive rule induced by the simulator.

Under this fixed-query convention, the posterior predictive can be decomposed over the semantic path as

$$P_0(y \mid x, D) = \int P_0(y \mid x, D, m)\, P_0(dm \mid D),$$

where $P_0(y \mid x, D, m)$ marginalizes any simulator latent variables not included in M. Adaptive-PFN *exposes* M by approximating the two factors in this integral separately. The semantic posterior network T_1 approximates $P_0(dm \mid D)$, while the conditional predictive network T_2 approximates $P_0(y \mid x, D, m)$.

Because the simulator supplies the realized semantic path during training, the two networks can be trained independently using proper scoring rules. When the corresponding densities are available, their log-score population objectives are

$$\mathcal{L}_1 = \mathbb{E}_{P_0}[-\log T_1(M \mid D)], \qquad \mathcal{L}_2 = \mathbb{E}_{P_0}[-\log T_2(y \mid x, D, M)],$$

whose population minimizers are $P_0(dm \mid D)$ and $P_0(dy \mid x, D, m)$, respectively. Appendix C gives the architectures and training objectives.[1]

The resulting training-prior Adaptive-PFN predictor is defined by

$$q_0(y \mid x, D) = \int T_2(y \mid x, D, m)\, T_1(dm \mid D). \tag{1}$$

If both networks equal their population targets, then $q_0(y \mid x, D) = P_0(y \mid x, D)$. The separation in Equation 1 isolates the semantic-path distribution that we allow to change at deployment.

2.2 A PRIOR SHIFT AT DEPLOYMENT TIME

Let $D_1^*, \ldots, D_T^*$ denote data histories from T deployment tasks, drawn from a deployment law P^*, and let $p^*(dm)$ denote the corresponding prior distribution of the semantic path M. We allow deployment to change the distribution of M while leaving the conditional data-generating mechanism given M unchanged.

To specify the allowed prior changes, fix a vector of prior-shift features $s(M) \in \mathbb{R}^{d_s}$ and a parameter set $\Theta \subset \mathbb{R}^{d_s}$. For $\eta \in \Theta$, define the exponential tilt

$$p_\eta(dm) = \exp\{\eta^\top s(m) - A(\eta)\}\, p_0(dm), \qquad A(\eta) = \log \mathbb{E}_{p_0}\left[\exp\{\eta^\top s(M)\}\right]. \tag{2}$$

Thus p_η changes the relative prior weights of semantic paths while retaining the support of the training prior; $\eta = 0$ recovers p_0.

[1] For continuous components of M, our implementation represents the conditional distribution implicitly and uses the energy score rather than the log score; see Appendix C.

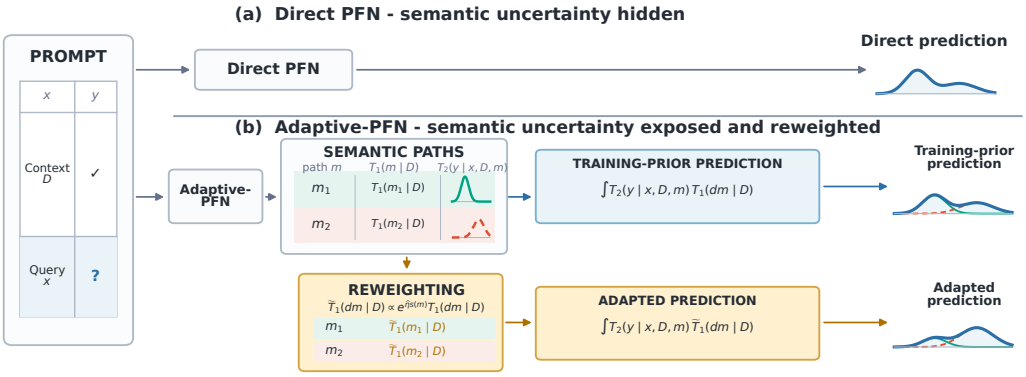

Figure 1: **Adaptive-PFN (sketch).** The context D contains observed input–response pairs; the query x has an unknown response y. For $y = Bax + \varepsilon$, the rows show representative positive- and negative-slope paths $m_1 = (+1, a_1)$ and $m_2 = (-1, a_2)$. **(a)** A direct PFN exposes only their combined prediction. **(b)** Adaptive-PFN exposes their posterior weights through T_1 and their conditional predictions through T_2. At deployment, $s(m) = B$ reweights the paths by $e^{\widehat{\eta} B}$ without updating either network.

Our prior-shift model assumes that, for some $\eta^* \in \Theta$,

$$p^*(dm) = p_{\eta^*}(dm), \qquad P^*(D, y \mid m, x) = P_0(D, y \mid m, x). \tag{3}$$

Hence deployment changes only the marginal distribution of the exposed semantic path; all remaining simulator variation is assumed invariant conditional on M. In the linear regression example of the introduction, taking $s(M) = B$ gives a one-dimensional tilt that changes the prior probability of positive versus negative slopes.

We shall let P_η denote the joint law induced by replacing p_0 with p_η while keeping this conditional law fixed, and let P_η^D denote its marginal law for the data history D.

2.3 Reweight an Allowed Prior Shift

First suppose $p_* = p_\eta$ for a known tilt η and Equation 3 holds. Bayes' rule then changes the semantic-path posterior by the prior ratio: $p_\eta(dm \mid D) \propto e^{\eta^\top s(m)} p_0(dm \mid D)$. Applying this ratio to T_1 and keeping T_2 fixed defines the *tilted Adaptive-PFN predictor*

$$q_\eta(y \mid x, D) = \frac{\int T_2(y \mid x, D, m) e^{\eta^\top s(m)} T_1(dm \mid D)}{\int e^{\eta^\top s(m)} T_1(dm \mid D)}. \tag{4}$$

If T_1 and T_2 equal their Bayesian targets, Equation 4 is the exact deployment predictive density. An independently estimated deployment prior that is absolutely continuous with respect to p_0 can also be supplied by replacing the exponential weight with its density ratio relative to p_0. In practice, the networks are approximate and η is unknown; we next estimate it and quantify the predictive cost of estimation. Appendix B derives the Bayes identity.

2.4 Estimate the Population Prior Tilt

We estimate η from the deployment data histories $D_1^*, \ldots, D_T^*$. For each data history D, $T_1(dm \mid D)$ gives a posterior distribution over its semantic path. Because the semantic path may remain uncertain, we do not select a single value of M. Instead, we average the prior-shift features under this posterior: $\bar{s}(D) = \int s(m) T_1(dm \mid D)$. In the running linear regression example, $\bar{s}(D) = T_1(B = +1 \mid D) - T_1(B = -1 \mid D)$. A value near 1 supports a positive slope, a value near -1 supports a negative slope, and a value near 0 reflects uncertainty about the sign.

From independent data histories $D_1^0, \ldots, D_N^0 \sim P_0^D$, let μ_0 be the empirical mean of $\bar{s}(D_1^0), \ldots, \bar{s}(D_N^0)$ and let W_0 be the inverse square root of their sample covariance. We define the *standardized semantic summary*

$$z(D) = W_0\{\bar{s}(D) - \mu_0\}. \tag{5}$$

Under the training distribution, $z(D)$ is centered at zero and its coordinates have comparable scales. Averaging $z(D)$ across deployment tasks gives a standardized estimate of how the population mean of the inferred prior-shift features differs from its value during training. In the running example, this average indicates whether the deployment population contains relatively more positive- or negative-slope tasks than the training population.

To predict this average under a candidate tilt, draw a reusable *source bank* $(M_b, D_b)_{b=1}^{N_0}$, consisting of N_0 independent *source tasks* generated by the training simulator under p_0. For bank task b, define the normalized weight $w_b(\eta) = \exp\{\eta^\top s(M_b)\}/\sum_{\ell=1}^{N_0}\exp\{\eta^\top s(M_\ell)\}$. We define the *estimated prior tilt* $\widehat{\eta}$ by

$$\widehat{\eta} = \arg\min_{\eta\in\Theta}\left\|\frac{1}{T}\sum_{t=1}^{T} z(D_t^*) - \sum_{b=1}^{N_0} w_b(\eta) z(D_b)\right\|_2^2. \tag{6}$$

This *whitened moment-matching* estimator matches the deployment mean summary to its reweighted source-bank estimate. Appendix B gives the details.

Our *adapted Adaptive-PFN predictor* is $q_{\widehat{\eta}}$: substitute the fitted $\widehat{\eta}$ into Equation 4. This changes semantic-path weights but leaves T_1, T_2, and their conditional predictions frozen.

We evaluate the population estimate while holding both networks fixed: how much predictive log loss is incurred by using $\widehat{\eta}$ instead of the true deployment tilt η_*? For a new deployment task, independent of the data histories used to fit $\widehat{\eta}$, let $p_{\eta_*}(\cdot \mid x, D)$ be its true predictive density. Using Kullback–Leibler (KL) divergence, define the *predictive log-loss regret*

$$\mathcal{R}(\eta) = \mathbb{E}_{D\sim P_{\eta_*}^D}\left[\mathrm{KL}(p_{\eta_*}(\cdot \mid x, D) \,\|\, q_\eta(\cdot \mid x, D))\right]. \tag{7}$$

It is the expected excess negative log-likelihood relative to the true deployment predictive distribution. Changing the tilt by v adds $v^\top s(m)$ to the log weight of semantic path m. Adding the same value to every path leaves the normalized mixture unchanged. We therefore measure the prediction-relevant size of the change by its *log-weight spread*, $\omega_s(v) = \sup_m v^\top s(m) - \inf_m v^\top s(m)$, which records how differently the tilt reweights the semantic paths.

Theorem 1 (Prediction under an estimated prior tilt)**.** *Suppose deployment follows an allowed tilt $\eta_* \in \Theta$, $s(M)$ is bounded, and the fixed networks satisfy $\mathcal{R}(\eta_*) < \infty$. Then, for every fitted tilt $\widehat{\eta}$,*

$$\left|\sqrt{\mathcal{R}(\widehat{\eta})} - \sqrt{\mathcal{R}(\eta_*)}\right| \le \frac{\omega_s(\widehat{\eta} - \eta_*)}{2\sqrt{2}}. \tag{8}$$

Treat the map z as fixed and suppose further that Θ is compact, the source-bank tasks are independent and identically distributed (i.i.d.) draws from the source simulator, the deployment data histories are i.i.d. draws from $P_{\eta_*}^D$, and the two samples are independent. Define $\psi(\eta) = \mathbb{E}_{P_\eta^D}[z(D)]$ and assume that $\psi(\eta) = \psi(\eta_*)$ implies $\eta = \eta_*$ for every $\eta \in \Theta$. Then the moment estimator in Equation 6 satisfies $\mathcal{R}(\widehat{\eta}) \xrightarrow{p} \mathcal{R}(\eta_*)$ as $T, N_0 \to \infty$. If η_* is interior and $\nabla_\eta\psi(\eta_*)$ has full column rank, and if T_1 and T_2 equal their target Bayesian distributions, then $\mathcal{R}(\widehat{\eta}) = O_p(T^{-1} + N_0^{-1})$.

The finite-sample inequality allows approximate networks and any tilt estimator: it compares a frozen predictor using $\widehat{\eta}$ with the same predictor using the true tilt. Under the ideal conditions of the rate statement, $\mathcal{R}(\eta_*) = 0$, and Equation 8 gives the quadratic bound $\mathcal{R}(\widehat{\eta}) \le \omega_s(\widehat{\eta} - \eta_*)^2/8$. Appendix B proves the theorem and states the corresponding rate for approximate networks.

To evaluate Equation 4, draw $M^{(1)}, \ldots, M^{(S)}$ from $T_1(dm \mid D)$, evaluate T_2 once for each path, and set $\alpha_j = \exp\{\widehat{\eta}^\top s(M^{(j)})\}$ and $\widetilde{\alpha}_j = \alpha_j/\sum_{k=1}^S \alpha_k$. The Monte Carlo adapted prediction is

$$\widehat{q}_{\widehat{\eta}}(y \mid x, D) = \sum_{j=1}^{S} \widetilde{\alpha}_j T_2(y \mid x, D, M^{(j)}). \tag{9}$$

This is self-normalized importance sampling. We report the *effective sample size* (ESS), $\mathrm{ESS} = 1/\sum_{j=1}^S \widetilde{\alpha}_j^2$, which ranges from S under equal weights to one when a single path dominates. A small ESS indicates that the approximation depends on few sampled paths.

2.5 TESTING COMPATIBILITY WITH THE PRIOR-SHIFT MODEL

The reweighted prediction requires Equation 3 and $p_* = p_\eta$ for some tilt in Equation 2. The two conditions imply that deployment data histories follow P^D_η for some allowed tilt. We test this through a goodness-of-fit (GOF) test, with the following null hypothesis:

$$H_0 : \quad P^D_* = P^D_\eta \text{ for some } \eta \in \Theta. \tag{10}$$

The prior-shift features $s(M)$ specify which deployment changes Adaptive-PFN can represent. To detect changes outside this family, we map each possibly variable-length data history D to a fixed *observable test representation* $g(D) \in \mathbb{R}^{d_g}$, whose coordinates are chosen to detect departures that an allowed tilt cannot reproduce. For example, let $g(D) = \widehat{\gamma}_2$, where $\widehat{\gamma}_2$ is the fitted coefficient of x^2 in a regression of y on $(1, x, x^2)$ using the observations in D. An allowed tilt changes the frequency of positive and negative slopes but preserves linearity, whereas a nonlinear deployment relationship can change the distribution of $\widehat{\gamma}_2$.

For any prior tilt η, let P^g_η denote the distribution of $g(D)$ when $D \sim P^D_\eta$. The null hypothesis contains every $\eta \in \Theta$, so it does not specify a single distribution against which to compare the deployment population. The test must first fit one allowed tilt to the observed data histories. We denote this tilt by $\widetilde{\eta}$ and fit it using a rule fixed as part of the test. This test prior tilt is separate from the moment-matching estimate $\widehat{\eta}$ used to adapt predictions. Appendix D.1 derives a population-likelihood estimator; the experimental appendices state the estimator used in each test. Under the null, there exists some $\eta_0 \in \Theta$ such that

$$g(D^*_1), \ldots, g(D^*_T) \overset{\text{iid}}{\sim} P^g_{\eta_0}.$$

Since η_0 is unknown, we estimate it by $\widetilde{\eta}$ and use $P^g_{\widetilde{\eta}}$ as the fitted reference distribution. We approximate this distribution using an independent bank $(M_b, D_b)_{b=1}^{N_{\text{ref}}}$ generated under the training prior. The bank is reweighted using the same prior-ratio correction as in Equation 4, with $\widetilde{\eta}$ in place of $\widehat{\eta}$.

After fitting $\widetilde{\eta}$, we use maximum mean discrepancy (MMD) (Gretton et al., 2012) to measure the discrepancy between the observed deployment distribution of $g(D)$ and the reweighted source-bank distribution predicted by the fitted prior shift. For distributions P and Q, independent draws $U, U' \sim P$ and $V, V' \sim Q$, and a kernel k measuring similarity, MMD is defined by

$$\text{MMD}^2_k(P, Q) = \mathbb{E}k(U, U') + \mathbb{E}k(V, V') - 2\mathbb{E}k(U, V).$$

It compares similarity within the two populations with similarity between them. Here, P is the deployment distribution of $g(D)$, represented by the observed vectors $g(D^*_1), \ldots, g(D^*_T)$, and $Q = P^g_{\widetilde{\eta}}$ is the distribution of $g(D)$ under the fitted allowed tilt. The kernel is fixed as part of the test construction. Let $\widehat{\text{MMD}}^2_k$ denote the weighted empirical estimate computed from the observed vectors and the reweighted bank. Our observed test statistic is

$$G_{\text{obs}} := T\widehat{\text{MMD}}^2_k.$$

A larger G_{obs} indicates greater disagreement with the fitted prior-shift model. Appendix D gives the explicit definitions.

The prior tilt $\widetilde{\eta}$ is chosen after seeing the observed deployment population. Thus, G_{obs} measures disagreement with the fitted allowed tilt, rather than with a tilt fixed in advance. The null comparison must give every simulated population the same opportunity to select its own best-fitting tilt. We therefore use a *refitted parametric bootstrap*: simulate B_{boot} populations from $P^D_{\widetilde{\eta}}$, refit the prior tilt separately in each one, and recompute the statistic to obtain $G^{(1)}, \ldots, G^{(B_{\text{boot}})}$. The upper-tail rank of G_{obs} among them gives the Monte Carlo value p_{MC}. A small value means that the observed disagreement is unusually large under the fitted prior-shift model. Appendix D gives the full construction; Figure 4 there illustrates the refitted bootstrap.

We note that this diagnostic is scoped because it can detect only discrepancies represented by $g(D)$. Rejection thus provides evidence that no allowed tilt reproduces those aspects of the deployment data histories.

3 CONTROLLED VALIDATION

In this first experiment the simulator and deployment prior are known, so we can evaluate the method. We ask whether Adaptive-PFN matches the predictive accuracy of a direct PFN with approximately the same number of parameters under the training prior, whether an estimated prior tilt recovers the prediction obtained with the true shift, and whether the scoped compatibility test rejects allowed shifts at the nominal rate while detecting prespecified changes outside the allowed prior-shift family.

3.1 CONTROLLED DESIGN

We study signed sparse noisy linear-regression tasks with 20 input features. For one task, the data history is $D = \{(x_i, y_i)\}_{i=1}^{n}$, where $x_i \in \mathbb{R}^{20}$ is a vector of 20 input features and $y_i \in \mathbb{R}$ is its response. At prediction time, $x \in \mathbb{R}^{20}$ is a new query input and y is its unknown response. The semantic path is $M = (K, \mathcal{I}, B, U)$: $K \in \{1, \dots, 8\}$ is the number of active input features, $\mathcal{I} \subset \{1, \dots, 20\}$ contains their indices, $B \in \{-1, +1\}$ indicates whether the active features tend to have positive or negative effects, and $U \in [0, 1]$ controls the observation-noise variance. For example, $K = 2$ and $\mathcal{I} = \{3, 11\}$ make only input features 3 and 11 relevant. There are 527,898 possible discrete configurations of $(K, \mathcal{I}, B)$, together with the continuous coordinate U; Appendix E gives the exact data generating process.

The training prior has $p_0(B = +1) = 0.8$, while deployment has $p_*(B = +1) = 0.2$; all other distributions remain fixed. Choosing $s(M) = B$ represents this change as an allowed prior shift of the form in Section 2.2. We train T_1, T_2, and a direct PFN with approximately the same number of parameters on independent tasks from p_0 and never retrain again. We repeat the deployment experiment with three independently sampled sets of tasks. In each repetition, 128 data histories of length 32 estimate the shared tilt, and 512 new tasks evaluate prediction. We train the models with three random seeds and evaluate every trained model on every set of tasks, giving nine seed–task-set combinations. For each evaluation task, we predict using data histories of length 8 and 32. Our primary metric is mean squared error (MSE) between the observed responses and their predictive means; lower is better. We form confidence intervals by independently resampling the three training runs and the three deployment task sets, thereby accounting for both sources of variation.

3.2 PREDICTIVE RECOVERY UNDER PRIOR SHIFTS

One-coordinate shift. Using data histories of length 8, we first allow deployment to change only the effect-sign frequency through $s(M) = B$. Under the training prior, Adaptive-PFN and the direct PFN obtain MSEs of 2.1123 and 2.1160, respectively, showing that the factorized architecture matches the predictive accuracy of the direct PFN before the deployment shift. After deployment changes $p(B = +1)$ from 0.8 to 0.2, the unadapted Adaptive-PFN has MSE 2.2535 and the direct PFN 2.2642. Moment matching from Section 2.4 reduces the Adaptive-PFN's MSE to 2.1234. This is a 5.77% reduction relative to its unadapted prediction (95% CI $[5.27, 6.16]$) and a 6.22% reduction relative to the direct PFN (95% CI $[5.56, 6.87]$). Using the known deployment tilt gives an MSE of 2.1229, while the population-likelihood estimator gives 2.1250. Table 1 reports MSE and negative log-likelihood (NLL); Figure 2(a) shows the predictive recovery.

The MSE reduction depends on posterior uncertainty about B after one task. Increasing the data-history length from 8 to 32 reduces the moment-matching gain from 5.77% to 1.81%, because the longer history leaves less posterior uncertainty about B. When deployment instead follows the training prior, adapting with the estimated prior tilt increases MSE by only 0.10%.

Multidimensional shift. We next allow deployment to change two components of the semantic path: the effect sign B and the number K of active input features. In addition to changing $p(B = +1)$ from 0.8 to 0.2, deployment favors sparser tasks, with the expected number of active input features decreasing from 4.5 under the training prior to 2.5. Using data histories of length 8, population-likelihood adaptation reduces MSE by 8.44% relative to the unadapted Adaptive-PFN, whereas the known deployment tilt reduces it by 8.46%. Moment matching reduces MSE by 6.08%. These results use the same frozen T_1 and T_2 as the one-coordinate experiment; neither network is retrained. Table 2 reports the seed-wise MSE reductions and estimated tilts.

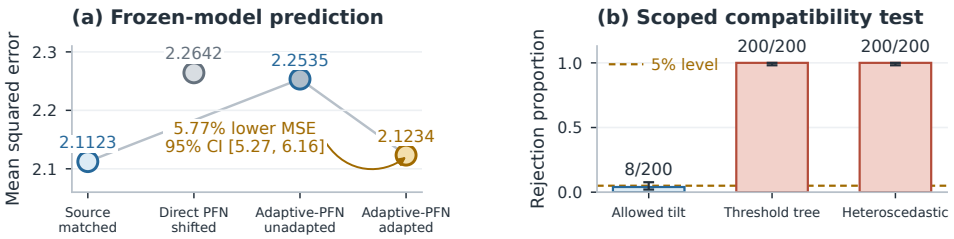

Figure 2: **Controlled recovery and scoped compatibility. (a)** Deployment changes $p(B{=}+1)$ from 0.8 to 0.2. Reweighting the frozen Adaptive-PFN reduces MSE by 5.77% relative to its unadapted prediction (95% CI [5.27, 6.16]). **(b)** At the 5% level, the test rejects 8/200 allowed shifts and 200/200 instances of each excluded mechanism; bars show 95% Wilson intervals.

3.3 Calibration and power

We first check calibration. At a nominal 5% level, the test should reject approximately 5% of populations generated by an allowed shift. Here, $s(M) = B$ allows the frequency of positive- and negative-effect tasks to change. For this test, $g(D)$ combines a general summary of the input–response distribution with summaries designed to detect nonlinear conditional means and input-dependent noise variance. It is constructed to be invariant to response-sign reversal, so changing the frequency of B does not alter its distribution, whereas the two excluded mechanisms exposed below can. Appendix E.5 gives the exact construction. Each evaluated population contains 128 data histories of length 32. The test rejects 8 of 200 compatible populations, a rate of 4.0% (95% Wilson interval $[2.0, 7.7]$). Appendix E.5 gives the complete protocol.

We next check power against two excluded changes. The threshold-tree mechanism makes the conditional mean depend nonlinearly on an active input; the heteroscedastic mechanism makes the noise variance input-dependent. The test rejects all 200 populations from each mechanism (95% Wilson interval $[98.1\%, 100\%]$); Figure 2(b) summarizes both results. The test is therefore calibrated under the allowed shift and detects these two departures. Its power remains specific to $g(D)$ and the departures considered here. Table 3 reports the calibration and power estimates, and Appendix E.5 gives their exact laws.

4 Synthetic-to-real forecasting

We now ask whether frozen models trained only on simulated tasks can adapt to real populations. We study electricity forecasting with the Low Carbon London data (Schofield et al., 2016) and retail-sales forecasting with the Rossmann Store Sales data (Cukierski, 2015). In both applications, the neural networks remain fixed. We measure prediction using weighted quantile loss (WQL; Appendix F.6); lower is better. Relative to the direct PFN, Adaptive-PFN reduces WQL by 31.4–45.3% on the London electricity data and by 7.48% on the Rossmann sales data.

4.1 Two forecasting tasks

London electricity. We use the Low Carbon London dynamic-tariff trial dataset. Each hour is assigned one of three tariff levels, called Normal, High, and Low. One task uses 48 preceding hours to predict consumption over the next 24 hours from the known tariff level, hour of day, and day of week. The training simulator allows the change in log consumption during High- and Low-tariff hours to vary across tasks and also varies trends, seasonality, persistence, and noise. Our primary metric is WQL evaluated on target hours with a High or Low tariff, which we call active hours. We train three seeds of each model; Appendix F gives the full task and simulator.

Rossmann retail sales. We use the Rossmann Store Sales dataset, which contains daily sales and schedules for two types of promotion: a promotion indicator for each day and Promo2, a recurring promotion program. One task uses 56 days from one store to predict the next 42 days. The training simulator allows the change in log sales under each promotion type to vary across tasks and also varies store levels, trends, seasonality, persistence, noise, and closures. For London, we estimate the deployment prior through T_1 using the moment-matching estimator in Section 2.4. For Rossmann,

we instead estimate it independently of T_1 through an empirical-Bayes procedure based on store-level regressions and supply the fitted prior to Adaptive-PFN (Appendix G.3). Its density ratio to the training prior p_0 then replaces the exponential tilt in Equation 4 (Appendix G.4).

To test whether adjusting the direct PFN's output is sufficient, we also include a *calibrated direct PFN*. It uses the same population-fitting data but changes only the predicted distribution. In London, it applies a tariff-specific affine correction to predicted consumption; in Rossmann, it adds promotion-specific corrections to predicted sales. Appendices F.4 and G.5 give the two constructions.

For the London electricity data, we also evaluate the compatibility test from Section 2.5. At the 5% level, validation rejects 5/60 training-prior populations, 0/60 allowed shifts, and 60/60 populations with an excluded dependence between the High- and Low-tariff associations that no allowed prior tilt can generate. The observed statistic G_{obs} lies below all 99 bootstrap values, so the test does not reject H_0 for the chosen $g(D)$ (Table 6; Appendix F.5).

4.2 Forecasting results

For the London electricity data, the calibrated direct PFN reduces WQL by 11.4–21.6% across seeds, and Adaptive-PFN reduces WQL by a further 12.5–38.2% relative to the calibrated direct PFN. Table 7 reports the seed-wise values, which Figure 3(a) displays. In 200 bootstrap replicates that resample both the population-fitting and evaluation households and re-estimate the prior tilt, the adapted Adaptive-PFN improves in every seed over both the unadapted Adaptive-PFN and the direct PFN (Table 8); Figure 3(b) visualizes these paired reductions. Negative log-likelihood (NLL), the average negative log predictive density, is also lower than for the direct PFN in every seed. The calibrated direct PFN lowers WQL but increases NLL; Adaptive-PFN lowers both. As a broader adaptation baseline, we also update all network parameters using the same deployment data. Early checkpoints attain lower WQL, but later ones overfit, so full-weight fine-tuning additionally requires checkpoint selection (Appendix F.7).

For the Rossmann sales data, Adaptive-PFN reduces WQL by 7.48% relative to the direct PFN across 3,840 open promotion days and also lowers NLL. Table 12 reports the comparison with all three frozen baselines.

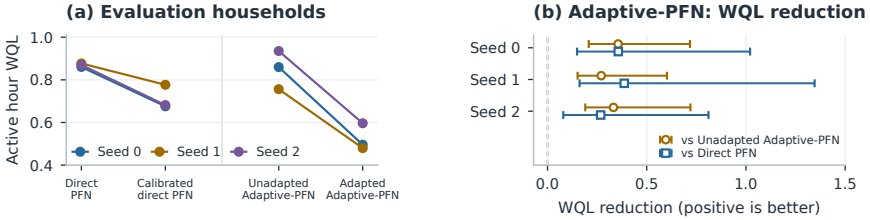

Figure 3: **London forecasting across three training seeds. (a)** Active-hour WQL for four frozen predictors. **(b)** Adapted Adaptive-PFN's WQL reduction relative to the direct PFN and unadapted Adaptive-PFN. 95% intervals resample both household sets and re-estimate the tilt.

5 Limitations and conclusion

PFNs amortize prediction under a fixed prior, but deployment populations change. Adaptive-PFN exposes the semantic-path mixture, enabling frozen networks to reweight the prior and test compatibility with deployment data. Across controlled shifts and two real applications, it improves prediction without retraining. Its limitations are prior support, the scope of $g(D)$, and repeated T_2 evaluations. Within this scope, it turns a fixed-prior PFN into an auditable, population-adaptive predictor.

Reproducibility statement

All code needed to reproduce the experiments are provided in the supplementary material accompanying this paper. A detailed construction of the used architectures, training procedures, numerical choices and experimental details can be found in the appendix.

ARTIFICIAL INTELLIGENCE USE STATEMENT

Generative artificial intelligence tools were used throughout the project as writing, coding, and organizational aids. They helped polish text drafted from preliminary notes and outlines supplied by the authors, who subsequently reviewed the full manuscript over several passes. They also helped locate relevant PFN, empirical-Bayes, and MMD literature; the authors subsequently checked every source and citation (Appendix A). They also helped identify the London electricity and Rossmann sales datasets as suitable real-world test beds for the methods developed here (Appendices F and G). The tools assisted with code for curating the experimental data, aggregating results, producing plots, and implementing and checking the synthetic training simulators against the data-generating processes specified by the authors (Appendices E, F, and G). During preparation of the appendix, they were especially useful for tracing implementation details across the codebase and consolidating them into a single description, including training settings, sample sizes, preprocessing for the datasets, and evaluation procedures. This assistance helped produce a more detailed and reproducible account of the experiments. Generative AI also suggested theoretical claims worth investigating, including the predictive-stability result formalized in Theorem 1, and helped draft its proof (Appendix B.4). The authors checked these claims and reviewed all protocols, mathematical arguments, reported results, and final text.

REFERENCES

Abdul Fatir Ansari, Oleksandr Shchur, Jaris Küken, Andreas Auer, Boran Han, Pedro Mercado, Syama Sundar Rangapuram, et al. Chronos-2: From univariate to universal forecasting, 2025.

Alexis Bellot and Mihaela van der Schaar. Application of kernel hypothesis testing on set-valued data. In *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pp. 194–204. PMLR, 2021.

Nick Cannella, Anzo Teh, Yanjun Han, and Yury Polyanskiy. Universal priors: Solving empirical Bayes via Bayesian inference and pretraining. In *Proceedings of Thirty Ninth Conference on Learning Theory*, volume 336 of *Proceedings of Machine Learning Research*, pp. 896–937. PMLR, 2026.

Paul Edmund Chang, Nasrulloh Ratu Bagus Satrio Loka, Daolang Huang, Ulpu Remes, Samuel Kaski, and Luigi Acerbi. Amortized probabilistic conditioning for optimization, simulation and inference. In *Proceedings of the 28th International Conference on Artificial Intelligence and Statistics*, volume 258 of *Proceedings of Machine Learning Research*, pp. 703–711. PMLR, 2025.

Will Cukierski. Rossmann store sales. Kaggle prediction competition, 2015. URL https://www.kaggle.com/competitions/rossmann-store-sales.

Samuel Dooley, Gurnoor Singh Khurana, Chirag Mohapatra, Siddartha V. Naidu, and Colin White. ForecastPFN: Synthetically-trained zero-shot forecasting. In *Advances in Neural Information Processing Systems*, volume 36, pp. 2403–2426, 2023. doi: 10.52202/075280-0112.

Lasse Elsemüller, Hans Olischläger, Marvin Schmitt, Paul-Christian Bürkner, Ullrich Köthe, and Stefan T. Radev. Sensitivity-aware amortized Bayesian inference. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856.

Saurabh Garg, Yifan Wu, Sivaraman Balakrishnan, and Zachary C. Lipton. A unified view of label shift estimation. In *Advances in Neural Information Processing Systems*, volume 33, pp. 3290–3300, 2020.

Tilmann Gneiting and Adrian E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007. doi: 10.1198/016214506000001437.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(25):723–773, 2012.

Léo Grinsztajn, Klemens Flöge, Oscar Key, Felix Birkel, Philipp Jund, Brendan Roof, et al. TabPFN-3: Technical report, 2026.

Kai Helli, David Schnurr, Noah Hollmann, Samuel Müller, and Frank Hutter. Drift-resilient TabPFN: In-context learning temporal distribution shifts on tabular data. In *Advances in Neural Information Processing Systems*, volume 37, pp. 98742–98781, 2024. doi: 10.52202/079017-3134.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *The Eleventh International Conference on Learning Representations*, 2023.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637(8045):319–326, 2025. doi: 10.1038/s41586-024-08328-6.

Benjamin Jäger, Nick Erickson, Léo Grinsztajn, Felix Birkel, Klemens Flöge, Oscar Key, et al. TabPFN-3.5: Technical report, 2026.

Ayush Jain and Rajat Sen. TimesFM-3: A zero-shot foundation model for multivariate forecasting. Google Research Blog, August 2026.

Oscar Key, Arthur Gretton, François-Xavier Briol, and Tamara Fernandez. Composite goodness-of-fit tests with kernels. *Journal of Machine Learning Research*, 26(51):1–60, 2025.

Weihao Kong and Abhimanyu Das. Introducing TabFM: A zero-shot foundation model for tabular data. Google Research Blog, June 2026.

Seunghan Lee. Mitigating label shift in tabular in-context learning via test-time posterior adjustment. In *Proceedings of the 43rd International Conference on Machine Learning*. PMLR, 2026.

Zachary Lipton, Yu-Xiang Wang, and Alexander Smola. Detecting and correcting for label shift with black box predictors. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3122–3130. PMLR, 2018.

Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do Bayesian inference. In *International Conference on Learning Representations*, 2022.

Samuel Müller, Matthias Feurer, Noah Hollmann, and Frank Hutter. PFNs4BO: In-context learning for Bayesian optimization. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 25444–25470. PMLR, 2023.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, volume 20, pp. 1177–1184. Curran Associates, Inc., 2007.

Arik Reuter, Tim G. J. Rudner, Vincent Fortuin, and David Rügamer. Can transformers learn full Bayesian inference in context? In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 51531–51582. PMLR, 2025.

Marco Saerens, Patrice Latinne, and Christine Decaestecker. Adjusting the outputs of a classifier to new a priori probabilities: A simple procedure. *Neural Computation*, 14(1):21–41, 2002. doi: 10.1162/089976602753284446.

Marvin Schmitt, Paul-Christian Bürkner, Ullrich Köthe, and Stefan T. Radev. Detecting model misspecification in amortized Bayesian inference with neural networks. *arXiv preprint arXiv:2112.08866*, 2021.

James R. Schofield, Richard Carmichael, Simon H. Tindemans, Mark Bilton, Matt Woolf, and Goran Strbac. Low carbon london project: Data from the dynamic time-of-use electricity pricing trial, 2013. UK Data Service, 2016. Study Number 7857.

P. A. V. B. Swamy. Efficient inference in a random coefficient regression model. *Econometrica*, 38 (2):311–323, 1970. doi: 10.2307/1913012.

Anzo Teh, Mark Jabbour, and Yury Polyanskiy. Solving empirical Bayes via transformers, 2025.

UK Power Networks. Smartmeter energy consumption data in london households. London Datastore, 2014. URL https://data.london.gov.uk/dataset/smartmeter-energy-consumption-data-in-london-households-vqm0d. Accessed 20 September 2026.

Maxime Vandegar, Michael Kagan, Antoine Wehenkel, and Gilles Louppe. Neural empirical Bayes: Source distribution estimation and its applications to simulation-based inference. In *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pp. 2107–2115. PMLR, 2021.

Julius Vetter, Guy Moss, Cornelius Schröder, Richard Gao, and Jakob H. Macke. Sourcerer: Sample-based maximum entropy source distribution estimation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 88772–88806. Curran Associates, Inc., 2024. doi: 10.52202/079017-2817.

Sebastian Wagner-Carena, Ji Won Park, Simon Birrer, Philip J. Marshall, Aaron Roodman, and Risa H. Wechsler. Hierarchical inference with Bayesian neural networks: An application to strong gravitational lensing. *The Astrophysical Journal*, 909(2):187, 2021. doi: 10.3847/1538-4357/abdf59.

George Whittle, Juliusz Ziomek, Jacob Rawling, and Maike A. Osborne. Distribution transformers: Fast approximate Bayesian inference with on-the-fly prior adaptation. In *Proceedings of the 43rd International Conference on Machine Learning*. PMLR, 2026.

Yang Yang, Severi Rissanen, Paul Edmund Chang, Nasrulloh Ratu Bagus Satrio Loka, Daolang Huang, Arno Solin, Markus Heinonen, and Luigi Acerbi. PriorGuide: Test-time prior adaptation for simulation-based inference. In *The Fourteenth International Conference on Learning Representations*, 2026.

Qingyang Zhu, Eric Karl Oermann, and Kyunghyun Cho. Multi-task Bayesian in-context learning. In *Proceedings of the 43rd International Conference on Machine Learning*. PMLR, 2026.

APPENDIX ROADMAP

Appendix	Section and contents	Page
A	**Related work** PFNs, test-time prior adaptation, population-prior estimation, and compatibility assessment	13
B	**Theory and proofs** Posterior reweighting, prior-tilt estimation, identification, overlap, and the predictive-adaptation theorem	15
C	**Adaptive-PFN implementation** The semantic posterior network T_1, conditional predictive network T_2, and Riemann predictive distribution	22
D	**Compatibility testing** Population-likelihood estimation, the observable discrepancy, the refitted bootstrap, and deployment under an estimated tilt	27
E	**Controlled experiment** Simulator, source training, evaluation, multidimensional prior shifts, and compatibility testing	32
F	**London electricity experiment** Forecasting tasks, source simulator, prior estimation, compatibility assessment, metrics, and ablations	39
G	**Rossmann sales experiment** Forecasting task, synthetic training distribution, deployment prior estimation, and evaluation	50

A RELATED WORK

Prior-Data Fitted Networks. Adaptive-PFN starts from the PFN construction. A Prior-Data Fitted Network (PFN) is trained on tasks sampled from a simulator and minimizes their expected predictive log loss (Müller et al., 2022). At the population optimum, its fixed weights implement the posterior predictive distribution induced by the simulator and its training prior. Training therefore teaches the network to approximate the Bayesian prediction rule. At inference time, a forward pass applies that rule to a new data history, avoiding separate posterior computation for every new task. This construction has enabled fast in-context prediction for tabular data (Hollmann et al., 2023; 2025), Bayesian optimization (Müller et al., 2023), and zero-shot forecasting (Dooley et al., 2023). PFN-style transformers can also be trained to return posterior samples rather than only the final predictive distribution (Reuter et al., 2025).

Recent tabular models apply this synthetic-pretraining strategy to a wider range of datasets. The second generation of TabPFN showed that a PFN trained on synthetic tables can provide competitive classification and regression on small tabular datasets (Hollmann et al., 2025). TabPFN-3 extends this strategy to larger tables and provides variants for time series, relational data, and tables containing text (Grinsztajn et al., 2026). TabPFN-3.5 further considers temporal and grouped splits, as well as wider and multimodal tables (Jäger et al., 2026). In parallel, TabFM treats tabular classification and regression as zero-shot in-context prediction after pretraining on a large collection of synthetic tables (Kong & Das, 2026). These models broaden the tasks that a frozen pretrained network can address. Adaptive-PFN considers a different problem: after training, a population of deployment tasks may reveal that the semantic paths occur at different frequencies than they did under the training prior. We ask how a declared part of that prior can be revised without retraining the network.

Recent time-series foundation models likewise use large-scale pretraining to forecast new series without task-specific training. Chronos-2 supports univariate, multivariate, and covariate-informed forecasting through in-context learning across related series (Ansari et al., 2025). TimesFM-3 supports zero-shot multivariate forecasting and can condition on both historical covariates and covariates known over the prediction horizon (Jain & Sen, 2026). The latter capability is relevant to our forecasting setting because future tariff states and calendar variables are known when electricity consumption is predicted. These models seek broad zero-shot generalization from a fixed pretrained model. Our synthetic-to-real experiments address a different source of information: a population of

related deployment data histories reveals a shared change in semantic-path frequencies. Adaptive-PFN estimates that change as a prior tilt and uses it to reweight the exposed semantic-path mixture.

The training prior determines how often each data-generating choice appears during PFN training. After training, a direct PFN continues to combine those possibilities according to the frequencies encoded by that prior, even when their frequencies differ in a new population of tasks. One response is to anticipate the shift during training: Drift-Resilient TabPFN, for example, includes temporal shifts in its synthetic training prior (Helli et al., 2024). Our starting point is different. We retain a broad training prior, expose the PFN's distribution over a chosen semantic path, and use a population of deployment data histories to revise the frequencies of those semantic paths after training. The network parameters remain frozen; only the weights in the exposed semantic-path mixture change.

Test-time prior adaptation. Exposing a semantic path connects Adaptive-PFN to amortized inference methods that keep latent structure available after training. The Amortized Conditioning Engine explicitly represents latent variables and supports runtime conditioning on data, latent values, and priors (Chang et al., 2025). Sensitivity-aware amortized inference trains jointly over alternative prior and likelihood specifications, making their effect available for efficient sensitivity analysis (Elsemüller et al., 2024). These methods retain latent variables or model specifications as explicit inputs or outputs instead of integrating them out irreversibly in the final prediction. Adaptive-PFN does this for the semantic path: T_1 exposes its posterior distribution, while T_2 returns the predictive distribution conditional on one specified semantic path. Separating these two operations makes the semantic-path weights contributed by the training prior explicit and therefore reweightable.

A complementary line of work supplies new prior information at inference time. Distribution Transformers map a supplied prior distribution and dataset to an approximate posterior (Whittle et al., 2026). PriorGuide uses an approximation to a supplied target-to-training prior ratio to guide a pretrained diffusion model (Yang et al., 2026). Multi-Task Bayesian In-Context Learning represents prior information through a prefix of related tasks and learns hierarchical predictive inference across prior families (Zhu et al., 2026). In these approaches, the new prior information enters respectively as a supplied distribution, an approximate prior ratio, or a prefix of related tasks. Adaptive-PFN does not require the deployment prior to be supplied, nor does it update its network parameters. It estimates a shared low-dimensional prior tilt from related deployment data histories, applies the corresponding prior-ratio correction to the semantic-path distribution returned by T_1, and averages the unchanged conditional predictive distributions returned by T_2.

Population prior estimation. Estimating the prior tilt from multiple deployment tasks places our method in the empirical-Bayes tradition. Empirical Bayes uses a collection of related datasets to estimate a shared population-level distribution and then uses that estimate when making inferences within each dataset. In simulation-based settings, Neural Empirical Bayes estimates simulator source distributions (Vandegar et al., 2021), while Sourcerer estimates a source distribution from samples using a maximum-entropy criterion (Vetter et al., 2024). Hierarchical amortized inference instead reuses individual posterior approximations to infer population-level parameters (Wagner-Carena et al., 2021). Transformers have also been trained to solve empirical-Bayes problems in context, including work that establishes theoretical guarantees for universal pretraining priors in the Poisson model (Teh et al., 2025; Cannella et al., 2026).

Our deployment data histories play the role of the related datasets, and the prior tilt parameterizes their shared population change. The population-likelihood estimator in Appendix D.1 is the direct empirical-Bayes estimator for this model. The moment-matching estimator used in the main text takes a more targeted route: it averages the prior-shift features under the semantic-path distribution returned by T_1, whitens this posterior summary under the training distribution, and finds the prior tilt whose reweighted source population matches its deployment mean. Estimating the population prior is only the first step. Adaptive-PFN checks whether the fitted tilt is supported by the source bank through the effective sample size, then applies it to the frozen semantic-path mixture to obtain a deployment prediction.

This update is also related to prior-probability adjustment in classification. Under label shift, the frequencies of the classes may change while the data law conditional on each class remains fixed. Posterior class probabilities can then be corrected by reweighting them according to the change in class frequencies (Saerens et al., 2002; Lipton et al., 2018; Garg et al., 2020). DistPFN applies

test-time posterior adjustment to label shift in tabular in-context learning (Lee, 2026). Our prior-shift assumption has the same conditional-invariance structure, but a class label is replaced by a structured semantic path. The semantic path may contain categorical and continuous coordinates, and its weights combine entire path-conditional predictive distributions rather than class probabilities.

Compatibility testing. After fitting a prior tilt, we ask whether it reproduces the features of the deployment data histories encoded by the prespecified observable test representation $g(D)$. This question is related to misspecification diagnostics for amortized Bayesian inference, which ask whether an observed dataset resembles data generated by the training simulator (Schmitt et al., 2021). Our question is narrower. We do not test the training distribution alone; the null contains every deployment distribution produced by an allowed prior tilt. A deployment population can therefore differ substantially from the training population and remain compatible with the adaptation model.

We compare populations through maximum mean discrepancy, a kernel-based measure of the difference between two distributions. Kernel tests can compare distributions of set-valued observations (Bellot & van der Schaar, 2021), which is useful here because each task contributes a data history rather than a single scalar observation. The remaining difficulty is that the prior tilt is unknown and is fitted from the same deployment population used to compute the test statistic. Treating the fitted tilt as fixed would omit this fitting step from the null comparison. We instead use the refitted parametric-bootstrap construction studied for kernel composite goodness-of-fit tests by Key et al. (2025): each bootstrap population receives the same opportunity to fit its own prior tilt before its discrepancy is computed.

Finally, the compatibility check is deliberately scoped by the observable test representation $g(D)$. Its coordinates are chosen before evaluation to capture departures that matter for the deployment and that the allowed prior-shift features cannot express. Rejection therefore says that no allowed prior tilt explains those retained behaviors. Non-rejection supports that specific adaptation model; it does not certify the full simulator or claim power against every possible departure.

B Theory and proofs

This section develops the formal argument behind Adaptive-PFN. The argument follows the same order as the deployment procedure in the main text. We first write the training distribution of one task and show how an allowed prior shift reweights its semantic-path posterior while leaving the conditional data law fixed. We then explain why the standardized semantic summary is informative about this change, derive the whitened moment-matching estimator, and separate identification from computational overlap. Finally, we prove Theorem 1, which connects error in the estimated prior tilt to predictive log-loss regret. Throughout this section, T_1 and T_2 remain fixed. The only deployment quantity being estimated is the prior tilt.

B.1 Posterior reweighting

The training simulator generates one task in the order described in Section 2.1. It first draws a semantic path M from the training prior p_0. Conditional on that semantic path, it generates the data history D and then the target y at a fixed query input x. Writing m for one possible value of M, the resulting joint distribution is written below in measure notation. The distinction between m and dm is important: m denotes one particular semantic path, while $p_0(dm)$ denotes the probability element assigned by the training-prior measure as M ranges over possible paths. Likewise, $P(dD \mid m)$ and $P(dy \mid x, D, m)$ assign probability over possible data histories and target values. Thus, dm, dD, and dy do not represent additional variables or ordinary derivatives; they indicate the arguments over which the corresponding probability measures range. If all variables had ordinary densities, we could instead write a joint density at (m, D, y) with respect to $dm\, dD\, dy$. We use measure notation because it also covers the discrete, continuous, and mixed semantic paths considered here without changing the expression. The resulting joint distribution is

$$P_0(dm, dD, dy \mid x) = p_0(dm)P(dD \mid m)P(dy \mid x, D, m). \tag{11}$$

This factorization makes the prior-shift assumption precise. An allowed deployment change replaces only p_0 by the tilted prior

$$p_\eta(dm) = \exp\{\eta^\top s(m) - A(\eta)\}\, p_0(dm), \qquad A(\eta) = \log \int \exp\{\eta^\top s(m)\}\, p_0(dm). \tag{12}$$

Here, η is the prior tilt, $s(m)$ contains the prespecified prior-shift features, and $A(\eta)$ is the log normalizer that makes p_η a probability distribution. This is the same tilted prior introduced in Equation 2. The data law within each semantic path—the last two factors in Equation 11—remains fixed. Thus, adaptation does not change what a semantic path means or how it predicts. It changes only how frequently the semantic paths occur.

After observing one data history, the same change can be expressed as a correction to the training-prior semantic-path posterior. Bayes' rule gives

$$p_\eta(dm \mid D) = \frac{e^{\eta^\top s(m)} p_0(dm \mid D)}{\int e^{\eta^\top s(m')} p_0(dm' \mid D)}. \tag{13}$$

The normalizing factor $e^{-A(\eta)}$ in Equation 12 cancels between the numerator and denominator because it does not depend on m. What remains is the path-specific factor $e^{\eta^\top s(m)}$. A semantic path favored by the prior tilt receives more weight; a path that is not favored receives less.

The deployment posterior predictive distribution averages the unchanged data law $P(dy \mid x, D, m)$ using the reweighted semantic-path posterior in Equation 13. Replacing these two population quantities by the semantic-path distribution returned by T_1 and the conditional predictive density returned by T_2 gives Equation 4. This establishes the exact Bayes identity behind Adaptive-PFN. Under the prior-shift assumption, approximation enters because the trained networks may differ from their Bayesian targets, the prior tilt must be estimated, and the semantic-path mixture is evaluated with finitely many samples.

B.2 Prior Estimation

B.2.1 Semantic Summaries

The semantic paths of deployment tasks are not observed, so their changing frequencies cannot be counted directly. The data histories are observed, and T_1 converts each one into a distribution over the semantic path. We now show why the posterior mean $\bar{s}(D) = \int s(m) T_1(dm \mid D)$ used in Section 2.4 is a natural observable summary of a prior shift.

Let $p_\eta^D(D)$ denote a density of the data-history distribution P_η^D. Correspondingly, let $p(D \mid m)$ denote a density of the fixed data-history law $P(dD \mid m)$ used in Equation 11. Marginalizing out the unobserved semantic path under Equation 12 gives

$$p_\eta^D(D) = e^{-A(\eta)} \int p(D \mid m) e^{\eta^\top s(m)} p_0(dm). \tag{14}$$

This expression makes the two contributions of η visible. The exponential inside the integral changes the relative weight of each semantic path, while $A(\eta)$ normalizes the tilted prior. Assuming differentiation may be passed under the integral, taking logarithms and differentiating gives

$$\nabla_\eta \log p_\eta^D(D) = \frac{\int s(m) p(D \mid m) e^{\eta^\top s(m)} p_0(dm)}{\int p(D \mid m) e^{\eta^\top s(m)} p_0(dm)} - \nabla_\eta A(\eta). \tag{15}$$

At the training prior, $\eta = 0$. By Bayes' rule, the ratio of integrals is then the posterior expectation of $s(M)$ given D. Moreover, differentiating the log normalizer in Equation 12 gives

$$\nabla_\eta A(\eta) = \frac{\int s(m) e^{\eta^\top s(m)} p_0(dm)}{\int e^{\eta^\top s(m)} p_0(dm)} = \mathbb{E}_{p_\eta}[s(M)]. \tag{16}$$

The first equality follows from the derivative of a logarithm and the chain rule. The ratio is an expectation under p_η because its denominator normalizes the exponentially weighted training prior. At $\eta = 0$, every exponential weight equals one and $p_\eta = p_0$. Since $\int p_0(dm) = 1$, Equation 16 reduces to

$$\nabla_\eta A(0) = \int s(m) p_0(dm) = \mathbb{E}_{p_0(M)}[s(M)]. \tag{17}$$

Hence

$$\nabla_\eta \log p_\eta^D(D)\big|_{\eta=0} = \mathbb{E}_{p_0(M|D)}[s(M)] - \mathbb{E}_{p_0(M)}[s(M)]. \tag{18}$$

The left-hand side is the score of one data history for the prior tilt: it measures how the log probability of the observed data history initially changes as the deployment prior moves away from the training prior. The first term on the right is what the observed data history implies about the prior-shift features, while the second is their baseline mean under the training prior. Their difference is therefore the update induced by that data history.

This identity is a property of the prior-shift model and does not itself depend on a learned network. The exactness of T_1 matters because it lets us evaluate the first term. If $T_1(dm \mid D) = p_0(dm \mid D)$, then $\bar{s}(D) = \mathbb{E}_{p_0(M|D)}[s(M)]$. Under this exactness condition, $\mu_0 = \mathbb{E}_{P_0^D}[\bar{s}(D)] = \mathbb{E}_{p_0}[s(M)]$, and the centered posterior mean $\bar{s}(D) - \mu_0$ is exactly the score of the data-history distribution for the prior tilt at $\eta = 0$. Thus, $\bar{s}(D) - \mu_0$ is the score contributed by one data history at $\eta = 0$. Averaging it across deployment tasks gives the empirical mean score of the deployment data histories. This score identity motivates the summary; it does not claim that $\bar{s}(D)$ is sufficient for every possible deployment change.

Different prior-shift features may nevertheless produce posterior means with different scales, and two features may produce strongly correlated movements. Matching their raw means would then give more influence to a coordinate merely because it has larger training variance or uses different units. We remove this dependence using the training covariance $C_0 = \mathrm{Cov}_{P_0^D}\{\bar{s}(D)\}$. Let $V\Lambda V^\top$ denote the part of its eigendecomposition corresponding to eigenvalues above a fixed numerical tolerance. We use

$$W_0 = \Lambda^{-1/2} V^\top, \qquad z(D) = W_0\{\bar{s}(D) - \mu_0\}. \tag{19}$$

The retained coordinates of the standardized semantic summary $z(D)$ have mean zero and identity covariance under the training distribution. Whitening therefore puts the retained directions on a common scale and discards directions whose eigenvalues are below the declared numerical tolerance. The training mean μ_0, retained eigenspace, and matrix W_0 are estimated once from training simulations and then held fixed during deployment.

B.2.2 Moment matching

The *deployment mean summary* $\bar{z}_T = T^{-1}\sum_{t=1}^T z(D_t^*)$ is observable. To estimate the prior tilt, we need a reference value for it under every candidate prior tilt. If a candidate value of η were known, we could generate many new tasks from p_η, compute their standardized semantic summaries, and average them. Repeating that simulation for every value considered during optimization would be wasteful. The source bank provides the same comparison from one reusable set of tasks generated under the training prior.

To see why this reuse is valid, let $f(D)$ be any integrable vector-valued function. Under a candidate prior tilt, its expectation can first be computed by conditioning on the semantic path and then averaging over the tilted prior:

$$\begin{aligned}
\mathbb{E}_{P_\eta^D}[f(D)] &= \int \left\{ \int f(D) P(dD \mid m) \right\} p_\eta(dm) \\
&= e^{-A(\eta)} \int \left\{ \int f(D) P(dD \mid m) \right\} e^{\eta^\top s(m)} p_0(dm) \\
&= \frac{\mathbb{E}_{P_0}[e^{\eta^\top s(M)} f(D)]}{\mathbb{E}_{p_0}[e^{\eta^\top s(M)}]}.
\end{aligned} \tag{20}$$

The first equality averages the fixed data law $P(dD \mid m)$ over semantic paths drawn from p_η. The second substitutes the definition of the tilted prior. The last equality uses $e^{A(\eta)} = \mathbb{E}_{p_0}[e^{\eta^\top s(M)}]$ and recognizes the remaining integral as an expectation under the joint training distribution of (M, D). Thus, no part of the data law is changed: only the contribution of each training-prior task is changed according to its semantic path.

The source bank replaces the two expectations in Equation 20 by sample averages. For its tasks $(M_b, D_b)_{b=1}^{N_0}$, their ratio is

$$\frac{N_0^{-1}\sum_{b=1}^{N_0} e^{\eta^\top s(M_b)} f(D_b)}{N_0^{-1}\sum_{b=1}^{N_0} e^{\eta^\top s(M_b)}} = \sum_{b=1}^{N_0} w_b(\eta) f(D_b). \tag{21}$$

At $\eta = 0$, every source-bank task has the same weight. For a nonzero candidate prior tilt, tasks with larger $\eta^\top s(M_b)$ receive more weight and tasks with smaller $\eta^\top s(M_b)$ receive less. The weights still sum to one, so the result remains an average rather than changing scale.

Setting $f = z$ gives the reweighted source estimate $\sum_{b=1}^{N_0} w_b(\eta) z(D_b)$. As η varies, the same source bank therefore traces the deployment mean summary predicted by each candidate prior tilt. Equation 6 selects the estimated prior tilt whose reweighted source estimate is closest to the observed deployment mean summary.

B.2.3 Optimization

Equation 6 defines the estimated prior tilt as the minimizer of the finite-bank objective

$$J_{T,N_0}(\eta) = \left\| \bar{z}_T - \sum_{b=1}^{N_0} w_b(\eta) z(D_b) \right\|_2^2, \qquad \eta \in \Theta. \tag{22}$$

All quantities other than η are computed once before optimization. The deployment mean summary $\bar{z}_T$, source-bank summaries $z(D_b)$, and prior-shift features $s(M_b)$ remain fixed. Evaluating a new candidate prior tilt therefore requires only recomputing the normalized source-bank weights and their weighted average. It requires no new simulation and no further evaluation of either network.

For numerical stability, we evaluate the weights with a shifted softmax. If $a_b(\eta) = \eta^\top s(M_b)$ and $a_{\max}(\eta) = \max_b a_b(\eta)$, then

$$w_b(\eta) = \frac{\exp\{a_b(\eta) - a_{\max}(\eta)\}}{\sum_{c=1}^{N_0} \exp\{a_c(\eta) - a_{\max}(\eta)\}}. \tag{23}$$

Subtracting the same maximum from every log weight leaves the normalized weights unchanged but prevents overflow when a candidate prior tilt produces large log weights.

The controlled fits in Appendix E have moment maps that can be inverted directly. In the one-coordinate prior shift, let $\widehat{p}$ be the fitted probability of $B = +1$. Since $p_0(B = +1) = 0.8$, its corresponding tilt is

$$\widehat{\eta}_B = \frac{1}{2}\left\{ \log\frac{\widehat{p}}{1-\widehat{p}} - \log\frac{0.8}{0.2} \right\}.$$

The two-dimensional prior shift uses the same expression for its B coordinate. Its other coordinate tilts the standardized number K of active input features. We find that coordinate by 100 bisection steps on $[-4, 4]$, using the monotone source expectation of standardized K. These fits therefore solve their moment equations without a generic optimizer.

For the multidimensional London fit, we optimize Equation 22 in double precision over the fixed box $\Theta = [-80, 80]^4$. The wide box prevents unbounded numerical excursions along weakly identified coordinates; because it still constrains the fit, we report below when a coordinate reaches its boundary. We use deterministic multistart L-BFGS with a strong-Wolfe line search. The starting values of η are the origin and the positive and negative unit coordinate directions, giving $2d_s + 1$ starts. Each is mapped to the unconstrained starting value $r = \operatorname{arctanh}(\eta/80)$. To keep every iterate inside the box, we then optimize r through $\eta = 80\tanh(r)$. The objective and its gradient are evaluated by automatic differentiation. Each start is allowed at most 200 L-BFGS iterations, with internal gradient and objective-change tolerances of 10^{-9} and 10^{-12}, respectively. We retain the start with the smallest value of J_{T,N_0} and report whether it passes the optimality check below.

Convergence is checked in the original η coordinates rather than only in the unconstrained r coordinates. Writing Π_Θ for Euclidean projection onto the parameter box, the reported certificate is the projected-gradient norm

$$\left\| \eta - \Pi_\Theta\{\eta - \nabla_\eta J_{T,N_0}(\eta)\} \right\|_2. \tag{24}$$

A fit satisfies the numerical optimality check when this norm is at most 10^{-6}. A coordinate on the boundary is reported separately. It is not automatically treated as an optimization failure, because a constrained optimum can legitimately lie on the boundary; it is instead interpreted together with the Jacobian singular values and source-bank effective sample size below.

The optimization is deterministic conditional on its inputs, but those inputs still contain three approximations. The semantic posterior network approximates $p_0(M \mid D)$, a finite set of training simulations estimates the training mean and whitening transformation, and the finite source bank approximates the reweighted population expectation. We use independent simulation streams for these roles and repeat the London fit with an independent source bank. Appendix F reports this check. Increasing their sizes can improve numerical approximation, but it cannot make a deployment change excluded by the prior-shift features representable by an allowed prior tilt.

B.3 Identification and overlap

Equation 6 finds the prior tilt whose reweighted source estimate matches the deployment mean summary. However, this match may not determine a unique prior tilt, and it may rely on only a small number of source-bank tasks. We therefore examine identification and overlap separately.

Different prior tilts can be distinguished only if they produce different expected standardized semantic summaries. Writing $\psi(\eta) = \mathbb{E}_{P_\eta^D}[z(D)]$, local identification requires the Jacobian $\nabla_\eta \psi(\eta)$ to have full column rank at the data-generating value. Small singular values indicate combinations of tilt coordinates that are only weakly informed by the matched mean. This is a local condition and does not rule out a distant prior tilt with the same population mean. We use it as a theoretical identification diagnostic rather than a quantity reported systematically in every experiment.

We also report whether the estimated prior tilt lies on the boundary of Θ. A boundary value can reflect weak identification or a population change more extreme than the prespecified range; it is a diagnostic, not by itself an optimization failure or evidence against the prior-shift assumption.

Identification is distinct from overlap. An identified prior tilt may still concentrate nearly all importance weight on a few source-bank tasks or sampled semantic paths. We measure these two sources of concentration separately.

The source-bank weights $w_1(\eta), \ldots, w_{N_0}(\eta)$ in Section 2.4 are normalized to sum to one. Their *source-bank effective sample size* is

$$\mathrm{ESS}_{\mathrm{bank}}(\eta) = \frac{1}{\sum_{b=1}^{N_0} w_b(\eta)^2}. \tag{25}$$

The ratio $\mathrm{ESS}_{\mathrm{bank}}/N_0$ lies between $1/N_0$ and one. It equals one when every source-bank task has the same weight and approaches $1/N_0$ when one task dominates. It therefore records how many source-bank tasks meaningfully support the reweighted source estimate used to fit the prior tilt.

For one prediction, let $\alpha_1, \ldots, \alpha_S$ be the unnormalized posterior-path weights defined after Equation 4. The *posterior-path effective sample size* is

$$\mathrm{ESS}_{\mathrm{path}} = \frac{(\sum_{j=1}^{S} \alpha_j)^2}{\sum_{j=1}^{S} \alpha_j^2}. \tag{26}$$

It records how many semantic paths sampled from $T_1(dm \mid D)$ meaningfully contribute to the adapted prediction for that task. This quantity can vary across data histories: adequate source-bank overlap does not guarantee adequate posterior-path overlap for every prediction.

The Jacobian singular values, source-bank effective sample size, and posterior-path effective sample size therefore diagnose different failure modes. None tests whether an allowed prior tilt actually explains the deployment population. That is the separate role of the compatibility test.

B.4 Proof of Theorem 1

The theorem separates statistical estimation from prediction. Its first part shows that, under identification, the whitened moment-matching estimator approaches the true deployment tilt and gives

its stochastic rate. Its second part is deterministic: for any two prior tilts, it bounds how much the predictive log-loss regret can change when the semantic-path weights change. Combining them carries the estimation rate through the same frozen predictor.

Proof. **Consistency and rate of the estimated prior tilt.** Write

$$\bar{z}_T = \frac{1}{T}\sum_{t=1}^{T} z(D_t^*), \qquad \psi_{N_0}(\eta) = \sum_{b=1}^{N_0} w_b(\eta) z(D_b).$$

The first quantity is the deployment mean summary. The second is the reweighted source estimate under a candidate prior tilt. Equation 20 expresses $\psi(\eta)$ as a ratio of training-distribution expectations. Because s is bounded and z is a fixed linear transformation of an expectation of s, both are bounded. On compact Θ, uniform laws of large numbers therefore give $\sup_{\eta\in\Theta} \|\psi_{N_0}(\eta) - \psi(\eta)\|_2 \to 0$ in probability. Because $D_1^*, \ldots, D_T^*$ are i.i.d. draws from $P_{\eta_*}^D$, the deployment mean summary is an empirical average under the true deployment tilt. The law of large numbers gives

$$\bar{z}_T = \frac{1}{T}\sum_{t=1}^{T} z(D_t^*) \xrightarrow{p} \mathbb{E}_{P_{\eta_*}^D}[z(D)] = \psi(\eta_*). \tag{27}$$

The global identification assumption in Theorem 1 states that $\psi(\eta) = \psi(\eta_*)$ only when $\eta = \eta_*$. Consequently, the population objective $\|\psi(\eta_*) - \psi(\eta)\|_2^2$ is zero only at η_* and therefore has η_* as its unique minimizer. The finite-bank objective converges uniformly to this population objective, so the argmin theorem gives $\widehat{\eta} \xrightarrow{p} \eta_*$.

For the rate, suppose η_* is interior and $J_* = \nabla_\eta \psi(\eta_*)$ has full column rank. Let the two centered sampling errors be

$$\xi_T = \bar{z}_T - \psi(\eta_*), \qquad \xi_0 = \psi_{N_0}(\eta_*) - \psi(\eta_*).$$

Independent sampling and boundedness give $\xi_T = O_p(T^{-1/2})$ and $\xi_0 = O_p(N_0^{-1/2})$. The exponential weights make ψ continuously differentiable, and the empirical derivatives converge uniformly. Expanding the first-order condition for Equation 6 around η_* gives

$$\widehat{\eta} - \eta_* = (J_*^\top J_*)^{-1} J_*^\top (\xi_T - \xi_0) + o_p(T^{-1/2} + N_0^{-1/2}). \tag{28}$$

The two leading errors have a direct interpretation: ξ_T is sampling error from the finite sample of deployment data histories, while ξ_0 is sampling error from the finite source bank. Thus $\widehat{\eta} - \eta_* = O_p(T^{-1/2} + N_0^{-1/2})$, and both sample sizes contribute to the rate.

Predictive stability between two tilts. Fix two tilts η_0 and η_1, set $v = \eta_1 - \eta_0$, and define $h(m) = v^\top s(m)$. Recall that the log-weight spread induced by this change is

$$\omega_s(v) = \sup_m v^\top s(m) - \inf_m v^\top s(m). \tag{29}$$

It measures the largest difference, across semantic paths, in how the change from η_0 to η_1 modifies their log weights. Let $\eta_t = \eta_0 + tv$ for $t \in [0, 1]$. For a fixed data history D, define the tilted learned semantic-path distribution

$$\pi_t(dm \mid D) = \frac{e^{\eta_t^\top s(m)} T_1(dm \mid D)}{\int e^{\eta_t^\top s(m')} T_1(dm' \mid D)}.$$

The path η_t moves continuously from η_0 to η_1, and $\pi_t(dm \mid D)$ is the semantic-path distribution returned by T_1 after reweighting at the intermediate tilt. After also observing a possible target value y, Bayes' rule within the learned mixture gives

$$\pi_t(dm \mid x, D, y) = \frac{T_2(y \mid x, D, m)\pi_t(dm \mid D)}{q_{\eta_t}(y \mid x, D)}.$$

This second distribution is used only to compare the semantic-path weights before and after conditioning on y; it is not another component of Adaptive-PFN. Differentiating the numerator and denominator of Equation 4 along the path of tilts gives

$$g_{t,D}(y) := \frac{d}{dt} \log q_{\eta_t}(y \mid x, D) = \mathbb{E}_{\pi_t(\cdot \mid x, D, y)} h(M) - \mathbb{E}_{\pi_t(\cdot \mid D)} h(M). \tag{30}$$

The derivative is the difference between two expectations of the same log-weight change $h(M)$. Consequently, as a function of y, $g_{t,D}$ has range at most $\omega_s(v)$. It also has mean zero under $q_{\eta_t}(\cdot \mid x, D)$ because it is the score of a normalized predictive density.

For one data history, let

$$K_D(t) = \mathrm{KL}(p_{\eta_*}(\cdot \mid x, D) \,\|\, q_{\eta_t}(\cdot \mid x, D)), \qquad F(t) = \mathbb{E}_{D\sim P^D_{\eta_*}}[K_D(t)] = \mathcal{R}(\eta_t). \tag{31}$$

Here and below, expectations over D are under $P^D_{\eta_*}$. The true deployment predictive distribution remains fixed along this path; only q_{η_t} changes. Because s is bounded, the ratio between $q_{\eta_t}(\cdot \mid x, D)$ and $q_{\eta_*}(\cdot \mid x, D)$ is bounded above and below by positive constants. Thus, finite $\mathcal{R}(\eta_*)$ implies finite $F(t)$ throughout the segment. Moreover, Equation 30 and the bounded range of h imply that $g_{t,D}$ is bounded, which justifies differentiation under the expectation. In the expectations below, Y denotes the target random variable with the distribution indicated by the expectation subscript. Differentiating Equation 31 gives

$$F'(t) = -\mathbb{E}_D \mathbb{E}_{p_{\eta_*}(\cdot|x,D)}[g_{t,D}(Y)]. \tag{32}$$

The score has mean zero under its own predictive distribution:

$$\mathbb{E}_{q_{\eta_t}(\cdot|x,D)}[g_{t,D}(Y)] = 0. \tag{33}$$

We can therefore subtract this zero expectation inside Equation 32. For each D, the range of $g_{t,D}$ is at most $\omega_s(v)$. With $\mathrm{TV}(P, Q) = \sup_A |P(A) - Q(A)|$, the range characterization of total variation gives

$$\left|\mathbb{E}_{p_{\eta_*}(\cdot|x,D)}[g_{t,D}(Y)] - \mathbb{E}_{q_{\eta_t}(\cdot|x,D)}[g_{t,D}(Y)]\right| \le \omega_s(v)\,\mathrm{TV}(p_{\eta_*}(\cdot \mid x, D), q_{\eta_t}(\cdot \mid x, D)). \tag{34}$$

Averaging this bound over data histories, applying Pinsker's inequality to each $K_D(t)$, and then applying Jensen's inequality gives

$$\begin{aligned} |F'(t)| &\le \omega_s(v)\,\mathbb{E}_D\,\mathrm{TV}(p_{\eta_*}(\cdot \mid x, D), q_{\eta_t}(\cdot \mid x, D)) \\ &\le \omega_s(v)\,\mathbb{E}_D\sqrt{K_D(t)/2} \le \omega_s(v)\sqrt{F(t)/2}. \end{aligned} \tag{35}$$

The log-weight spread $\omega_s(v)$ limits how differently the two tilts can favor semantic paths, while $F(t)$ measures the current discrepancy from the true deployment predictive distribution. For every $\epsilon > 0$, Equation 35 implies

$$\left|\frac{d}{dt}\sqrt{F(t) + \epsilon}\right| \le \frac{\omega_s(v)}{2\sqrt{2}}.$$

Integrating from zero to one and taking $\epsilon \downarrow 0$ yields

$$\left|\sqrt{\mathcal{R}(\eta_1)} - \sqrt{\mathcal{R}(\eta_0)}\right| \le \frac{\omega_s(\eta_1 - \eta_0)}{2\sqrt{2}}. \tag{36}$$

Setting $\eta_0 = \eta_*$ and $\eta_1 = \widehat{\eta}$ proves Equation 8.

Combining the two parts. Since bounded s gives $C_s := \sup_{\|v\|_2=1} \omega_s(v) < \infty$ and hence $\omega_s(v) \le C_s\|v\|_2$, consistency and Equation 28 give

$$\mathcal{R}(\widehat{\eta}) \xrightarrow{p} \mathcal{R}(\eta_*), \qquad |\mathcal{R}(\widehat{\eta}) - \mathcal{R}(\eta_*)| = O_p\Big(\sqrt{\mathcal{R}(\eta_*)}(T^{-1/2} + N_0^{-1/2}) + (T^{-1/2} + N_0^{-1/2})^2\Big). \tag{37}$$

When the trained networks differ from their Bayesian targets, $\mathcal{R}(\eta_*)$ is the regret of the same frozen predictor supplied with the true deployment tilt. When T_1 and T_2 equal their target Bayesian distributions and the semantic-path mixture is evaluated without Monte Carlo error, $\mathcal{R}(\eta_*) = 0$. The remaining error is then quadratic in the tilt estimation error, and Equation 37 gives $\mathcal{R}(\widehat{\eta}) = O_p(T^{-1} + N_0^{-1})$. □

The deterministic stability bound also applies to the implemented finite mixture when the same sampled semantic paths are reused at both tilts. The final $O_p(T^{-1} + N_0^{-1})$ rate, however, assumes that T_1 and T_2 equal their target Bayesian distributions and that the semantic-path mixture is evaluated without Monte Carlo error. It does not give a separate rate in the number of sampled semantic paths.

C ADAPTIVE-PFN IMPLEMENTATION

Each simulated source task supplies a data history $D = \{(x_i, y_i)\}_{i=1}^n$, its semantic path M, and target responses at query inputs. T_1 learns the distribution of M given D, while T_2 learns the predictive distribution conditional on D, a query input, and the true M. The direct PFN instead receives the same data histories and query inputs as T_2, but not M.

C.1 SEMANTIC POSTERIOR NETWORK

T_1 is trained from pairs (D, M) to approximate the training-prior posterior $p_0(dm \mid D)$. To evaluate and sample this distribution, the implementation fixes the order $(M_1, \ldots, M_L)$ of the L choices that can form a semantic path. Here, M_ℓ denotes the ℓth possible choice and m_ℓ denotes its realized value when that choice is present. Each choice may be categorical or continuous, and it need not appear in every semantic path. We use $m_{1:\ell-1}$ for the realized values of the choices that are present before M_ℓ. Whether M_ℓ appears, and which values it may take, can depend on this preceding part of the path.

The training simulator specifies these restrictions before either network is trained. Given the preceding choices $m_{1:\ell-1}$, let

$$a_\ell(m_{1:\ell-1}) \in \{0, 1\}, \qquad \mathcal{S}_\ell(m_{1:\ell-1}). \tag{38}$$

Here, $a_\ell(m_{1:\ell-1})$ indicates whether M_ℓ is part of the semantic path. If $a_\ell = 1$, the set $\mathcal{S}_\ell(m_{1:\ell-1})$ contains the values that M_ℓ is allowed to take: a finite set for a categorical choice and $[0, 1]$ for a continuous choice. If $a_\ell = 0$, M_ℓ is absent from that semantic path, and T_1 neither predicts it nor includes a loss for it. Thus, a_ℓ and $\mathcal{S}_\ell$ describe the semantic paths permitted by the training simulator.

T_1 has the chain-rule factorization

$$T_1(dm \mid D) = \prod_{\ell:\, a_\ell(m_{1:\ell-1})=1} T_{1,\ell}(dm_\ell \mid D, m_{1:\ell-1}). \tag{39}$$

The factor $T_{1,\ell}$ is a probability mass function on $\mathcal{S}_\ell(m_{1:\ell-1})$ for a categorical choice and a distribution on $[0, 1]$ for a continuous choice. As in Appendix B, dm_ℓ indicates the variable over which the probability measure ranges.

The computation of one active factor $T_{1,\ell}$ is

$$\begin{aligned} D = \{(x_i, y_i)\}_{i=1}^n &\longrightarrow (h_1^D, \ldots, h_n^D) \longrightarrow H_D = \mathrm{HistoryEncoder}(h_1^D, \ldots, h_n^D), \\ (\ell, m_{1:\ell-1}, \log n, \epsilon_\ell) &\longrightarrow r_\ell^{(0)}, \\ (r_\ell^{(0)}, H_D) &\longrightarrow r_\ell = \mathrm{CrossAttn}_{\times 2}(r_\ell^{(0)}, H_D) \longrightarrow T_{1,\ell}(\cdot \mid D, m_{1:\ell-1}). \end{aligned} \tag{40}$$

Here, H_D contains the encoded data-history tokens, $r_\ell^{(0)}$ is the initial query for choice M_ℓ, and r_ℓ is the same query after two cross-attention blocks. The noise vector ϵ_ℓ appears only for a continuous choice.

Data-history encoding. For each data-history row, T_1 forms

$$(x_i, y_i) \longrightarrow h_i^D = E_D[x_i; y_i] \longrightarrow H_D, \tag{41}$$

Each observed pair (x_i, y_i) is represented by one token, with $[x_i; y_i]$ denoting the concatenation of its input and response. They are represented together because evidence about the semantic path comes from their relationship: T_1 must determine which data-generating choices are consistent with the observed responses at the corresponding inputs. The learned linear map E_D projects the concatenated vector into the common token space.

Four pre-normalized transformer encoder blocks apply self-attention across these tokens, followed by layer normalization. This allows the representation of each observation to depend on the other observations in D, rather than encoding every row independently. Their output

$$H_D = (\widetilde{h}_1^D, \ldots, \widetilde{h}_n^D) \tag{42}$$

retains one contextualized token for each observed row. It is not reduced to a single pooled vector: the subsequent cross-attention blocks can attend directly to the individual observations when predicting each semantic-path choice.

No positional encoding is added because the rows of D are treated as a set. Any ordering information required by an application must therefore be contained in the inputs x_i; in the forecasting applications, these inputs include time features. Data histories within the same training batch may contain different numbers of observations. To store them in one tensor, shorter histories are extended with placeholder rows. These rows are masked as attention keys in both the history encoder and the subsequent cross-attention blocks, so they cannot affect valid data tokens or semantic-path queries.

Semantic-path queries. The initial query which will cross-attend to H_D is:

$$r_\ell^{(0)} = \underbrace{e_\ell}_{\text{current choice}} + \underbrace{E_n(\log n)}_{\text{history length}} + \underbrace{\mathrm{LN}\left\{ \sum_{r<\ell:\, a_r(m_{1:r-1})=1} E_r(m_r) \right\}}_{\text{preceding active choices}} + \underbrace{E_\epsilon(\epsilon_\ell)}_{\text{continuous choice only}} . \tag{43}$$

$r_\ell^{(0)}$ specifies what T_1 should infer from the encoded data history. Its four terms identify the choice being predicted, provide the number of observations in D, represent the preceding part of the semantic path, and, when the choice is continuous, provide the noise used to generate one draw.

The learned embedding e_ℓ identifies M_ℓ, the semantic-path choice currently being predicted; it does not encode the unknown value m_ℓ. The term $E_n(\log n)$ is a learned representation of the number of observations in the data history, allowing the predicted distribution to depend explicitly on how much data is available. The prefix term tells T_1 which semantic choices have already been made and what values they took. This is needed because the distribution of the current choice M_ℓ may depend on the preceding choices $M_1, \ldots, M_{\ell-1}$. Each preceding choice M_r has its own encoder E_r, which maps its realized value m_r into the common hidden space. For a categorical choice, E_r is a learned embedding table; for a continuous choice, it is a learned linear map applied to $2m_r - 1$. The resulting vectors $E_r(m_r)$ are summed and layer-normalized to form a single representation of the realized prefix. When $\ell = 1$, there are no preceding choices, so this prefix term is omitted.

When M_ℓ is continuous, the query also includes $E_\epsilon(\epsilon_\ell)$, where $\epsilon_\ell \sim \mathcal{N}(0, I_{16})$ and E_ϵ is a learned linear map. The continuous output head uses this noise to produce a draw: changing ϵ_ℓ while holding D and the preceding choices fixed can produce a different value of M_ℓ. No noise term is used for a categorical choice because its output head returns the probabilities of all permitted categories directly.

The two cross-attention blocks then combine the initial query with the encoded data history H_D. The query identifies what is being predicted and the preceding choices on which the prediction is conditioned, while H_D represents the observed data. In each block, the query attends to the data-history tokens as keys and values. The resulting representation r_ℓ is passed to the appropriate output head. Each block uses pre-normalized multi-head attention followed by a GELU feed-forward layer, with a residual connection around each operation.

Conditional output. The output of the second cross-attention block is passed to the output head for choice ℓ. The form of this head depends on whether the choice is categorical or continuous:

$$r_\ell \longrightarrow \begin{cases} \mathrm{MaskedSoftmax}(W_\ell\, \mathrm{LN}(r_\ell)), & M_\ell \text{ categorical}, \\ 10^{-4} + (1 - 2\times 10^{-4})\ \mathrm{sigmoid}\big\{\mathrm{MLP}_{\mathrm{cont}}\big(\mathrm{LN}(r_\ell)\big)\big\}, & M_\ell \text{ continuous}. \end{cases} \tag{44}$$

If M_ℓ is categorical, the learned linear map W_ℓ converts the normalized query into one real-valued score for each category that can occur at choice ℓ. These scores are not yet probabilities. The preceding choices may rule out some categories. Their scores are therefore excluded according to $\mathcal{S}_\ell(m_{1:\ell-1})$, and the softmax converts the remaining scores into probabilities that sum to one. The result is the conditional probability mass function $T_{1,\ell}(\,\cdot \mid D, m_{1:\ell-1})$.

If M_ℓ is continuous, one evaluation of the network instead returns one value. The shared linear–GELU–linear map $\mathrm{MLP}_{\mathrm{cont}}$ reduces the normalized query to a scalar, the sigmoid maps that scalar into $(0, 1)$, and the final affine transformation restricts the returned value to $(10^{-4}, 1 - 10^{-4})$. This margin keeps generated values strictly away from the endpoints of the unit interval. Because the noise vector ϵ_ℓ is part of r_ℓ, evaluating the head with different noise vectors produces different draws while holding D and $m_{1:\ell-1}$ fixed. The continuous conditional distribution is therefore represented through samples rather than by an explicit density.

Training T_1. For every source task, the training simulator provides both the data history D and the realized semantic path m that generated it. Training proceeds through the choices in their fixed order. For each choice ℓ, let $\overline{\mathcal{L}}_\ell$ be its loss averaged over the tasks in the training batch for which that choice is present. If no task in a batch contains choice ℓ, that choice is omitted from the batch loss. When T_1 predicts M_ℓ, it receives the realized prefix $m_{1:\ell-1}$ rather than values predicted by T_1 itself:

$$(D, m) \longrightarrow \big\{T_{1,\ell}(\cdot \mid D, m_{1:\ell-1}) : a_\ell(m_{1:\ell-1}) = 1\big\} \longrightarrow \mathcal{L}_{T_1} = \sum_{\ell=1}^{L} \overline{\mathcal{L}}_\ell. \tag{45}$$

Using the true preceding values ensures that each factor is trained for the conditional distribution in Equation 39, without errors from earlier predictions entering its training input. Choices that are absent from the semantic path are skipped and contribute no loss.

For a categorical M_ℓ, the loss is the negative log-probability assigned to its true realized value m_ℓ:

$$\mathcal{L}_\ell^{\text{cat}} = -\log T_{1,\ell}\big(m_\ell \mid D, m_{1:\ell-1}\big). \tag{46}$$

Minimizing this loss encourages the categorical head to assign high probability to the value selected by the simulator.

For a continuous M_ℓ, the network is trained through samples because its output head does not return an explicit density. Holding D and $m_{1:\ell-1}$ fixed, we draw J independent noise vectors and obtain continuous outputs $V_1, \ldots, V_J$. If u is the true value of M_ℓ, the loss for that choice is the scalar energy score (Gneiting & Raftery, 2007):

$$\frac{1}{J}\sum_{j=1}^{J} |V_j - u| - \frac{1}{J(J-1)} \sum_{1 \le j < k \le J} |V_j - V_k|. \tag{47}$$

The first term penalizes draws that are far from the true value. The second term subtracts half the mean distance between pairs of draws, so the loss also accounts for the dispersion of the predicted distribution rather than rewarding only point predictions. Together, the two terms form a proper scoring rule: in population, its expected value is minimized when the distribution of the generated draws equals the true conditional distribution. We use $J = 16$ draws throughout. Increasing J reduces the Monte Carlo variation in this loss but increases the cost of training the continuous heads proportionally. The total loss sums the choice-specific averages over the L possible choices. Averaging before summing prevents choices that appear in more simulated tasks from receiving greater weight solely because they occur more often.

Deployment sampling. The true path is unavailable at deployment, so each complete path is generated from left to right. At choice ℓ, the sampled prefix first determines whether that choice is present. If it is, its value is drawn from the corresponding conditional distribution:

$$\begin{aligned} m_{1:0}^{(j)} &= \varnothing, \\ a_\ell(m_{1:\ell-1}^{(j)}) = 1 \quad &\Longrightarrow \quad m_\ell^{(j)} \sim T_{1,\ell}(\cdot \mid D, m_{1:\ell-1}^{(j)}), \\ a_\ell(m_{1:\ell-1}^{(j)}) = 0 \quad &\Longrightarrow \quad \text{skip choice } \ell, \qquad \ell = 1, \ldots, L. \end{aligned} \tag{48}$$

For a categorical choice that is present, the draw is restricted to $\mathcal{S}_\ell(m_{1:\ell-1}^{(j)})$. Generating one choice at a time therefore produces only valid semantic paths without enumerating all possible combinations of their values. Repeating the procedure produces the semantic-path samples used in Equation 9.

All experiments use hidden width 192, eight attention heads, feed-forward width 384, four data-history encoder layers, and two cross-attention blocks in T_1. We fixed this shared architecture before the repeated experiments and did not tune it separately for each application. The experiment appendices specify the semantic-path choices, their prefix restrictions, and the parameter counts.

C.2 Conditional predictive network

For a given semantic path m, T_2 learns the predictive distribution of a new response y conditional on the query input x, the observed data history D, and m. Each source task used to train T_2

therefore contains a data history D, its generating semantic path M, and query–response pairs $\{(x_j^{\mathrm{qry}}, y_j^{\mathrm{qry}})\}_{j=1}^{n_{\mathrm{qry}}}$, where n_{qry} is the number of query–response pairs. The training simulator supplies the true M, so T_2 learns the conditional distribution $T_2(y \mid x, D, m)$ from Section 2.1 without errors from T_1 entering its training inputs. At deployment, M is unknown, and T_2 is evaluated for the semantic paths sampled from $T_1(dm \mid D)$.

Semantic-path encoding. Our implementation supplies each semantic path to T_2 through a fixed-dimensional numerical vector. Before training, each experiment fixes a deterministic map $c : \mathcal{M} \to [-1, 1]^{d_M}$, where $\mathcal{M}$ is the set of valid semantic paths and d_M is the number of entries in their numerical representation. The map is injective, meaning that two different semantic paths cannot receive the same vector. Consequently, conditioning on $c(m)$ retains all the information in m. The map is fixed rather than learned by T_2; the experiment appendices specify its entries and their scaling.

Prompt construction. Write $c = c(m)$. The context and query tokens are constructed as

$$\begin{aligned} (x_i, y_i, c) &\longrightarrow h_i^{\mathrm{ctx}} = E_x(x_i) + E_y(y_i) + E_M c, \\ (x_j^{\mathrm{qry}}, c) &\longrightarrow h_j^{\mathrm{qry}} = E_x(x_j^{\mathrm{qry}}) + E_M c. \end{aligned} \tag{49}$$

Each observed pair (x_i, y_i) becomes a context token because both its input and response are available. A query token contains x_j^{qry} but not y_j^{qry}, which is the unknown response being predicted. The learned affine maps E_x and E_y place the inputs and observed responses in the common 192-dimensional token space. The learned linear map E_M represents the specified semantic path in the same space. Its output $E_M c$ is added to every context and query token, allowing the path to affect both the representation of the observed data and the prediction at the query. The prompt consists of all context tokens followed by all query tokens:

$$\mathcal{P}_m = \big(h_1^{\mathrm{ctx}}, \ldots, h_n^{\mathrm{ctx}}, h_1^{\mathrm{qry}}, \ldots, h_{n_{\mathrm{qry}}}^{\mathrm{qry}}\big). \tag{50}$$

Attention mask. Writing $\rightsquigarrow$ to mean that the token on the left may attend to the tokens on the right, the mask is

$$\begin{aligned} h_i^{\mathrm{ctx}} &\rightsquigarrow \{h_1^{\mathrm{ctx}}, \ldots, h_i^{\mathrm{ctx}}, \ldots, h_n^{\mathrm{ctx}}\}, \\ h_j^{\mathrm{qry}} &\rightsquigarrow \{h_1^{\mathrm{ctx}}, \ldots, h_n^{\mathrm{ctx}}, h_j^{\mathrm{qry}}\}, \\ h_j^{\mathrm{qry}} &\not\rightsquigarrow h_k^{\mathrm{qry}} \qquad (k \neq j). \end{aligned} \tag{51}$$

A context token attends to every observed row, allowing the transformer to represent relationships across the data history. It cannot attend to query tokens, so the representation of D does not depend on which queries are evaluated with it. Each query token attends to all context tokens and to itself, giving it access to the complete observed history. It cannot attend to another query token. Consequently, the prediction for x_j^{qry} cannot depend on which other queries are included in the same forward pass; grouping several queries together changes only the computational batching.

As in T_1, placeholder context rows are excluded by the attention mask. No positional encoding is added because the observed rows are treated as a set; any ordering information needed for prediction is contained in x_i.

Output probabilities. Six transformer layers apply the mask in Equation 51. Each layer contains multi-head self-attention and a GELU feed-forward sublayer, with residual connections and post-residual layer normalization. The encoded query token is then mapped to one real-valued score for each of the $R = 100$ response components defined in Section C.3:

$$\begin{aligned} \mathcal{P}_m &\longrightarrow (u_1^{\mathrm{qry}}, \ldots, u_{n_{\mathrm{qry}}}^{\mathrm{qry}}) = \mathrm{Transformer}_{\times 6}(\mathcal{P}_m), \\ u_j^{\mathrm{qry}} &\longrightarrow o_j = W_{\mathrm{out}} u_j^{\mathrm{qry}} + b_{\mathrm{out}} \longrightarrow \pi_j = \mathrm{softmax}(o_j), \qquad j = 1, \ldots, n_{\mathrm{qry}}. \end{aligned} \tag{52}$$

Here, u_j^{qry} is the final representation of query row j returned by the transformer, and W_{out} and b_{out} are the learned output matrix and bias. Their affine transformation produces the vector o_j of R real-valued scores. These scores are not yet probabilities. The softmax converts them into the vector $\pi_j = (\pi_{j0}, \ldots, \pi_{j,R-1})$ of nonnegative probability masses that sum to one. Section C.3 explains how these masses define the continuous predictive density $T_2(y \mid x_j^{\mathrm{qry}}, D, m)$.

Training T_2. During training, the simulator supplies the response y_j^{qry} associated with each query input x_j^{qry}. The true semantic path M is included in every prompt because the objective trains the predictive distribution conditional on that path. Writing $X_{\mathrm{qry}} = \{x_j^{\mathrm{qry}}\}_{j=1}^{n_{\mathrm{qry}}}$, training follows

$$\big(D, M, X_{\mathrm{qry}}\big) \longrightarrow \big\{T_2(\cdot \mid x_j^{\mathrm{qry}}, D, M)\big\}_{j=1}^{n_{\mathrm{qry}}} \longrightarrow \mathcal{L}_{T_2} = -\frac{1}{n_{\mathrm{qry}}} \sum_{j=1}^{n_{\mathrm{qry}}} \log T_2(y_j^{\mathrm{qry}} \mid x_j^{\mathrm{qry}}, D, M). \tag{53}$$

The value inside each logarithm is the density that the Riemann predictive distribution assigns to the observed query response. The negative logarithm is large when this density is small, so minimizing the average loss encourages T_2 to place high predictive density on the responses generated by the simulator. T_1 and T_2 have separate objectives and optimizers and receive independent streams of source tasks from the same training simulator.

Direct PFN baseline. The direct PFN provides a comparison with approximately the same number of parameters and the same source-task budget, but without an explicit semantic-path representation. It receives the same data histories and query inputs as T_2 but removes the semantic-path input from the token construction:

$$h_i^{\mathrm{ctx,dir}} = E_x^{\mathrm{dir}}(x_i) + E_y^{\mathrm{dir}}(y_i), \qquad h_j^{\mathrm{qry,dir}} = E_x^{\mathrm{dir}}(x_j^{\mathrm{qry}}). \tag{54}$$

The maps E_x^{dir} and E_y^{dir} are learned parameters of the direct PFN, not weights shared with T_2. The direct PFN uses the same attention mask, Riemann predictive distribution, and negative-log-likelihood objective as T_2. It has twelve transformer layers, and its feed-forward width is adjusted in each experiment so that its parameter count is approximately equal to the combined count of T_1 and T_2. The direct PFN replays the union of the independent T_1 and T_2 source-task streams, so its number of source training tasks matches their combined budget.

T_2 uses hidden width 192, eight attention heads, feed-forward width 384, and six transformer layers. Like the T_1 architecture, these dimensions were fixed before the repeated experiments and were not tuned separately for each application. All networks are initialized independently from prespecified seeds using the default PyTorch initialization for their constituent modules. They are trained with AdamW using $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, peak learning rate 3×10^{-4}, and weight decay 10^{-4}. The global gradient ℓ_2-norm is clipped at one. The learning rate increases linearly over the first $1/16$ of updates and then follows a cosine decay. Feed-forward layers use GELU activations and dropout is zero. The controlled and London training runs used PyTorch 2.9.1 with CUDA 12.8 on NVIDIA A100-SXM4-80GB graphics processing units; the Rossmann training run used PyTorch 2.4.1. We use the final checkpoint from each prespecified training budget; deployment outcomes do not select checkpoints or update network weights.

C.3 Riemann predictive distribution

The softmax in Equation 52 returns probability masses, not a density over the continuous response. The Riemann predictive distribution turns those masses into a density using R fixed bins (Müller et al., 2022). All experiments use $R = 100$.

Let $a_0 < a_1 < \cdots < a_R$ be fixed borders on the training-response scale after the preprocessing specified in the experiment appendix. Let ϕ and Φ denote the standard-normal density and cumulative distribution function, and define

$$\tau_L = \frac{a_1 - a_0}{\Phi^{-1}(0.75)}, \qquad \tau_R = \frac{a_R - a_{R-1}}{\Phi^{-1}(0.75)}. \tag{55}$$

The density of response component k is

$$b_k(y) = \begin{cases} \dfrac{2}{\tau_L}\phi\left(\dfrac{a_1 - y}{\tau_L}\right)\mathbf{1}\{y \le a_1\}, & k = 0, \\ \dfrac{\mathbf{1}\{a_k < y \le a_{k+1}\}}{a_{k+1} - a_k}, & 1 \le k \le R-2, \\ \dfrac{2}{\tau_R}\phi\left(\dfrac{y - a_{R-1}}{\tau_R}\right)\mathbf{1}\{y > a_{R-1}\}, & k = R-1. \end{cases} \tag{56}$$

Each b_k integrates to one. Therefore, if a predictive network returns $\pi = (\pi_0, \ldots, \pi_{R-1})$, its predictive density is

$$f_\pi(y) = \sum_{k=0}^{R-1} \pi_k b_k(y). \tag{57}$$

The interior components spread their mass uniformly over successive bounded intervals. The first and last components are half-normal densities extending to $-\infty$ and $+\infty$, so f_π has support on the full real line. The factor $\Phi^{-1}(0.75)$ makes a_0 the median of the left component and a_R the median of the right component. The points a_1 and a_{R-1} are where these tails meet the interior intervals.

Response bins. The borders are fixed before neural training using responses drawn independently from the training simulator. Let $y^{\text{grid}}_{(1)} \le \cdots \le y^{\text{grid}}_{(N_{\text{grid}})}$ be their ordered values on the training-response scale, and define

$$a_0 = y^{\text{grid}}_{(1)}, \qquad a_R = y^{\text{grid}}_{(N_{\text{grid}})}, \qquad a_k = \frac{y^{\text{grid}}_{(j_k)} + y^{\text{grid}}_{(j_k+1)}}{2}, \quad j_k = \left\lfloor \frac{kN_{\text{grid}}}{R} \right\rfloor, \quad 1 \le k < R. \tag{58}$$

Thus, each component contains approximately the same number of source responses used to construct the borders. If $y^{\text{grid}}_{(j_k)} = y^{\text{grid}}_{(j_k+1)}$ at a quantile split, the construction is rejected rather than perturbing the border. No deployment response is used to define the components, and every predictor and training seed within an experiment uses the same borders.

For the component probabilities π_j returned by T_2, Equation 57 gives $T_2(y \mid x_j^{\text{qry}}, D, m) = \sum_{k=0}^{R-1} \pi_{jk} b_k(y)$. The negative log-likelihood in Equation 53 therefore trains the complete predictive density rather than only a point prediction.

Mixture prediction. Draw S semantic paths $M^{(1)}, \ldots, M^{(S)}$ from $T_1(dm \mid D)$, and let $\pi_k^{(j)}$ be the mass that T_2 assigns to component k conditional on $M^{(j)}$. Let $\widehat{\rho}(m)$ be the estimated deployment-prior density relative to the training prior, up to a normalizing constant. The unnormalized weight of path $M^{(j)}$ is $\alpha_j = \widehat{\rho}(M^{(j)})$, and its normalized weight is $\widetilde{\alpha}_j = \alpha_j / \sum_{r=1}^{S} \alpha_r$. The adapted component mass is

$$\overline{\pi}_k = \sum_{j=1}^{S} \widetilde{\alpha}_j \pi_k^{(j)}. \tag{59}$$

This operation is exactly the finite semantic-path mixture in Equation 9: because every path-conditional prediction uses the same response components, their probabilities can be averaged component by component. The resulting adapted density is $\sum_{k=0}^{R-1} \overline{\pi}_k b_k(y)$. Under the exponential-tilt model in Section 2.3, one may take $\widehat{\rho}(m) = \exp\{\widehat{\eta}^\top s(m)\}$ because its common normalizing constant cancels when the weights are normalized. When the deployment prior is estimated directly, as in the Rossmann sales experiment, $\widehat{\rho}(m) = \widehat{p}_*(m)/p_0(m)$; Appendix G.4 gives that construction. We do not average neural logits, which would generally produce a different distribution because the softmax is nonlinear. Predictive means, quantiles, WQL, and negative log-likelihoods are all computed from this single mixed Riemann distribution.

D Compatibility Testing

The compatibility test examines the null hypothesis introduced in Equation 10:

$$H_0 : \quad P_*^D = P_\eta^D \text{ for some } \eta \in \Theta.$$

Here, P_*^D is the deployment distribution of data histories and P_η^D is the distribution induced by an allowed prior tilt η. Thus, the null permits any change represented by the family $\{P_\eta^D : \eta \in \Theta\}$ and excludes changes outside that family.

D.1 POPULATION-LIKELIHOOD ESTIMATOR

The compatibility null contains one data-history distribution P_η^D for every allowed prior tilt $\eta \in \Theta$. The test must therefore fit one member of this family before it can measure the discrepancy between the observed deployment population and the null. We denote the fitted value by $\widetilde{\eta}$ and call it the *test prior tilt*. It is distinct from the moment-matching estimate $\widehat{\eta}$ used to adapt predictions: $\widehat{\eta}$ is estimated by whitened moment matching, whereas $\widetilde{\eta}$ is used only to construct the reference distribution for the compatibility test.

Ideally, the test prior tilt would maximize the joint likelihood of the observed deployment data histories:

$$\widetilde{\eta} = \arg\max_{\eta \in \Theta} \sum_{t=1}^{T} \log p_\eta^D(D_t^*). \tag{60}$$

Here, $p_\eta^D(D)$ is the density of a data history under prior tilt η. This density is not directly available because the semantic path is unobserved. If $p(D \mid m)$ denotes the fixed conditional density of D given semantic path m, then

$$p_\eta^D(D) = \int p(D \mid m) p_\eta(dm).$$

Evaluating this expression directly would require integrating over all possible semantic paths.

The required likelihood can instead be expressed relative to the training distribution. Recall that

$$p_\eta(dm) = e^{\eta^\top s(m) - A(\eta)} p_0(dm), \qquad A(\eta) = \log \mathbb{E}_{p_0}\left[e^{\eta^\top s(M)}\right].$$

For any D with $p_0^D(D) > 0$, substituting the tilted prior into the marginal density and applying Bayes' rule gives

$$\begin{aligned} \frac{p_\eta^D(D)}{p_0^D(D)} &= \int e^{\eta^\top s(m) - A(\eta)} \frac{p(D \mid m) p_0(dm)}{p_0^D(D)} \\ &= \int e^{\eta^\top s(m) - A(\eta)} p_0(dm \mid D) \\ &= \mathbb{E}_{p_0(M \mid D)}\left[e^{\eta^\top s(M) - A(\eta)}\right]. \end{aligned} \tag{61}$$

The density ratio therefore depends on D through its training-prior semantic-path posterior. A path contributes in proportion to its posterior probability under $p_0(dm \mid D)$ and to the change $e^{\eta^\top s(m)}$ in its prior weight, while $e^{-A(\eta)}$ applies the common normalization required by the tilted prior.

For independent deployment data histories, summing the logarithm of this ratio gives

$$\sum_{t=1}^{T} \log p_\eta^D(D_t^*) = \sum_{t=1}^{T} \log p_0^D(D_t^*) + \mathcal{L}(\eta),$$

where

$$\mathcal{L}(\eta) = \sum_{t=1}^{T} \log \mathbb{E}_{p_0(M \mid D_t^*)}\left[e^{\eta^\top s(M)}\right] - T A(\eta). \tag{62}$$

The first sum does not depend on η, so maximizing the joint likelihood is equivalent to maximizing $\mathcal{L}(\eta)$. For each data history, the logarithmic expectation is the log average of the prior-weight multiplier $e^{\eta^\top s(M)}$ under the training-prior posterior for that history. The term $TA(\eta)$ accounts for the normalization of the tilted prior across the T histories.

The posterior expectation for each deployment history and the training-prior expectation defining $A(\eta)$ can both be approximated without evaluating p_η^D directly. For each deployment history D_t^*, draw $M_t^{(1)}, \ldots, M_t^{(S)}$ from $T_1(dm \mid D_t^*)$. These draws approximate the expectation under $p_0(dm \mid D_t^*)$. The N_0 source-bank paths $M_1, \ldots, M_{N_0}$ approximate

$$e^{A(\eta)} = \mathbb{E}_{p_0}\left[e^{\eta^\top s(M)}\right].$$

Substituting the two Monte Carlo approximations gives

$$\widehat{\mathcal{L}}(\eta) = \sum_{t=1}^{T} \log\left(\frac{1}{S}\sum_{j=1}^{S} e^{\eta^\top s(M_t^{(j)})}\right) - T\log\left(\frac{1}{N_0}\sum_{b=1}^{N_0} e^{\eta^\top s(M_b)}\right). \tag{63}$$

The resulting population-likelihood estimate is

$$\widetilde{\eta} = \arg\max_{\eta\in\Theta} \widehat{\mathcal{L}}(\eta). \tag{64}$$

This estimate is approximate because T_1 approximates the training-prior semantic-path posterior and both expectations are evaluated with finite samples. A compatibility test using this estimator takes $P^D_{\widetilde{\eta}}$ as its fitted reference distribution within the allowed family.

D.2 Test statistic

The compatibility null is stated in terms of the distribution of the complete data history D. Because D may be high-dimensional and variable in length, we deliberately restrict the test to a prespecified finite-dimensional observable test representation $g(D) \in \mathbb{R}^{d_g}$. Its coordinates retain observable properties that the fitted prior-shift model should reproduce and that can reveal changes outside the family generated by the prior-shift features $s(M)$.

The compatibility test compares the deployment and fitted-null distributions of $g(D)$ rather than the complete distributions of D. This deliberately scopes the diagnostic: rejection indicates that no allowed prior tilt reproduces the retained features of the deployment data histories, whereas non-rejection does not establish equality of the complete data-history distributions. The map g is fixed before the deployment population is examined and does not use T_1. Any source sample used to define g or the kernel is disjoint from the reference bank and the populations used to evaluate the test. Let k be the fixed kernel used to compare two values of $g(D)$. Each experimental appendix specifies its representation, any source-only standardization, and its kernel construction.

For any distribution P^D of data histories, write P^g for the induced distribution of $g(D)$ when $D \sim P^D$. In particular, P^g_* and P^g_η are induced by P^D_* and P^D_η, respectively. The population discrepancy is

$$\begin{aligned}\mathrm{MMD}^2_k(P^g_*, P^g_\eta) = \mathbb{E}_{D^*,\widetilde{D}^*\sim P^D_*}\left[k\Big(g(D^*), g(\widetilde{D}^*)\Big)\right] \\ - 2\,\mathbb{E}_{\substack{D^*\sim P^D_* \\ D\sim P^D_\eta}}\left[k(g(D^*), g(D))\right] \\ + \mathbb{E}_{D,\widetilde{D}\sim P^D_\eta}\left[k\Big(g(D), g(\widetilde{D})\Big)\right].\end{aligned} \tag{65}$$

The reference bank is generated under the training distribution rather than under P^D_η. Under the prior-shift model, expectations with respect to P^D_η can be obtained by reweighting source tasks according to

$$\rho_\eta(M) = \frac{dp_\eta}{dp_0}(M) = \exp\{\eta^\top s(M) - A(\eta)\}.$$

Let the reference bank contain N_{ref} tasks, with M_b denoting the semantic path that generated reference task D_b. The corresponding self-normalized importance weight for task b is

$$\widetilde{w}_b(\eta) = \frac{\exp\{\eta^\top s(M_b)\}}{\sum_{c=1}^{N_{\mathrm{ref}}}\exp\{\eta^\top s(M_c)\}}.$$

The factor $e^{-A(\eta)}$ in $\rho_\eta(M)$ is common to every reference task and therefore cancels when these weights are normalized. Writing $u_t = g(D^*_t)$ and $v_b = g(D_b)$, replacing the expectations in Equation 65 by their empirical counterparts gives the biased empirical squared MMD (Gretton et al., 2012)

$$\begin{aligned}\widehat{\mathrm{MMD}}^2_k(\eta) = \frac{1}{T^2}\sum_{t,t'=1}^{T} k(u_t, u_{t'}) - \frac{2}{T}\sum_{t=1}^{T}\sum_{b=1}^{N_{\mathrm{ref}}}\widetilde{w}_b(\eta)k(u_t, v_b) \\ + \sum_{b,b'=1}^{N_{\mathrm{ref}}}\widetilde{w}_b(\eta)\widetilde{w}_{b'}(\eta)k(v_b, v_{b'}).\end{aligned} \tag{66}$$

The observed statistic is

$$G_{\mathrm{obs}} = T\widehat{\mathrm{MMD}}_k^2(\widetilde{\eta}).$$

The factor T places the squared discrepancy on the scale used for its null calibration; because the same factor is applied to the observed and bootstrap statistics, it does not change their rank. If k is characteristic, the population MMD is zero exactly when the deployment and reference distributions of $g(D)$ agree. This does not imply equality of the complete data-history distributions.

D.3 Refitted bootstrap

The previous section defines the observed discrepancy after fitting the test prior tilt $\widetilde{\eta}$, namely

$$G_{\mathrm{obs}} = T\widehat{\mathrm{MMD}}_k^2(\widetilde{\eta}).$$

To turn this discrepancy into a test, we must determine how large G_{obs} would typically be when the compatibility null is true. This calibration must account for the fact that $\widetilde{\eta}$ is itself estimated from the same population on which the discrepancy is evaluated. If the tilt were treated as fixed in the null simulations, the observed population would have been allowed to choose its best-fitting tilt whereas the simulated populations would not. We therefore use a refitted parametric bootstrap: each simulated population is given the same opportunity to fit its own test prior tilt before its discrepancy is computed.

Figure 4 illustrates this calibration. The observed population and every simulated population first fit their own test prior tilt and then compute the corresponding discrepancy.

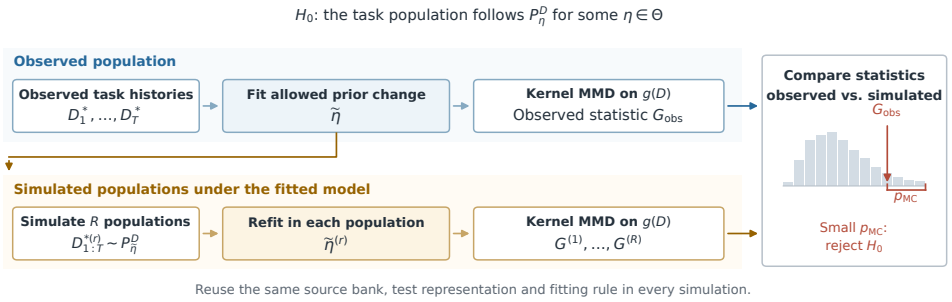

Figure 4: **Compatibility test when the prior tilt is unknown.** Fit the allowed prior change and compute G_{obs}. Then simulate R populations under the fitted model, refit in each, and compute $G^{(1)}, \ldots, G^{(R)}$. The upper-tail rank of G_{obs} gives p_{MC}; a small value rejects compatibility. The histogram is illustrative.

The fitting rule for the test prior tilt is fixed before the test is applied and is used unchanged for both the observed and simulated populations. Algorithm 1 summarizes the resulting procedure.

Algorithm 1 Refitted parametric bootstrap for the compatibility test

Require: Deployment histories $D_1^*, \dots, D_T^*$, number of bootstrap replicates B_{boot}, significance level α_{test}
Ensure: Monte Carlo value p_{MC} and compatibility decision
1: Fit the test prior tilt $\widetilde{\eta}$ to $D_1^*, \dots, D_T^*$ using the fixed fitting procedure
2: Compute $G_{\mathrm{obs}} = T\widehat{\mathrm{MMD}}_k^2(\widetilde{\eta})$
3: **for** $r = 1, \dots, B_{\mathrm{boot}}$ **do**
4: Simulate $D_1^{*(r)}, \dots, D_T^{*(r)} \sim P_{\widetilde{\eta}}^D$
5: Refit the test prior tilt to obtain $\widetilde{\eta}^{(r)}$
6: Compute $G^{(r)} = T\widehat{\mathrm{MMD}}_k^2(\widetilde{\eta}^{(r)})$
7: **end for**
8: $p_{\mathrm{MC}} \leftarrow \dfrac{1 + \sum_{r=1}^{B_{\mathrm{boot}}} \mathbf{1}\{G^{(r)} \geq G_{\mathrm{obs}}\}}{B_{\mathrm{boot}} + 1}$
9: **if** $p_{\mathrm{MC}} \leq \alpha_{\mathrm{test}}$ **then**
10: **return** reject the compatibility null
11: **else**
12: **return** do not reject the compatibility null
13: **end if**

The sum in the numerator counts the bootstrap statistics that are at least as large as G_{obs}. The additional one treats the observed statistic as one of the $B_{\mathrm{boot}} + 1$ values entering its own rank, so the smallest possible Monte Carlo value is $1/(B_{\mathrm{boot}} + 1)$ rather than zero. Using $G^{(r)} \geq G_{\mathrm{obs}}$ counts ties against rejection.

This rank calculation is exact in finite samples when the observed and simulated statistics are exchangeable under the null. That condition does not hold exactly here because the bootstrap populations are generated from $P_{\widetilde{\eta}}^D$, with $\widetilde{\eta}$ estimated from the observed deployment population. Refitting the test prior tilt within every bootstrap population reproduces the estimation step, but it does not restore exact finite-sample exchangeability. We therefore interpret p_{MC} as a refitted parametric-bootstrap value at nominal level α_{test}, not as an exact finite-sample p-value (Key et al., 2025). The experimental appendices evaluate its finite-sample Type-I error under the allowed prior-shift family.

Computational cost. Evaluating Equation 66 costs $O(T^2 + TN_{\mathrm{ref}} + N_{\mathrm{ref}}^2)$ kernel evaluations. The $N_{\mathrm{ref}} \times N_{\mathrm{ref}}$ reference matrix is computed once and reused. Each bootstrap replicate additionally simulates one population and refits its test prior tilt. When exact pairwise kernel evaluation is too costly, a fixed L-dimensional random Fourier feature (RFF) map ϕ (Rahimi & Recht, 2007) may approximate the kernel as $k(u, v) \approx \phi(u)^\top \phi(v)$. The empirical MMD then becomes the squared distance between the deployment and weighted-reference feature means, which costs $O\{(T+N_{\mathrm{ref}})L\}$ after the feature map has been evaluated. The RFF frequencies and phases are fixed before calibration and reused for the observed and bootstrap populations.

D.4 Test calibration and power

Let $\varphi(D_{1:T})$ be the indicator that the complete compatibility procedure rejects after observing a population of T data histories. Its auxiliary Monte Carlo randomness is suppressed from the notation; every probability below averages over that randomness as well as the observed population. If the population is generated by an allowed prior tilt η, the *Type-I error at* η is

$$\mathbb{P}_{(P_\eta^D)^{\otimes T}}\{\varphi(D_{1:T}) = 1\}.$$

This is the probability of rejecting even though the population belongs to the declared prior-shift family. It is a pointwise rejection probability for that value of η and does not by itself establish uniform error control over all of Θ.

Power must instead be defined relative to a particular departure from the allowed prior-shift family. Let Q denote the distribution of one data history under a specified excluded data-generating mechanism, and let Q^g denote the induced distribution of the selected representation $g(D)$ when $D \sim Q$. For a

population of T independent histories from Q, the power is

$$\mathbb{P}_{Q^{\otimes T}}\{\varphi(D_{1:T}) = 1\}.$$

The MMD discrepancy sees this departure through Q^g, whereas the complete rejection probability also depends on the procedure used to fit the test prior tilt. There is therefore no single power value: it changes with Q, the population size, the selected representation, the kernel, the significance level, the tilt-fitting rule, and the number of bootstrap replicates. If

$$Q^g = P_\eta^g \qquad \text{for some } \eta \in \Theta,$$

then the population MMD at that value of η is zero, even when $Q \neq P_\eta^D$. The discrepancy can detect only departures that change the distribution of $g(D)$ relative to the fitted reference distribution.

Non-rejection provides no evidence against the allowed prior-shift family in the selected representation at level α_{test}. It does not validate the complete simulator or imply that adaptation improves predictive performance.

E CONTROLLED EXPERIMENT

The controlled experiment evaluates prior estimation, prediction-time reweighting, and compatibility testing in a setting where the training prior, deployment prior, and data-generating mechanism are known. We first define the semantic path and the allowed prior shift. We then describe how T_1 and T_2 are trained under the training prior. With both networks frozen, a population of deployment data histories estimates the deployment prior, and this estimate reweights semantic paths when predicting on new tasks. Finally, the compatibility test asks whether an allowed prior shift is sufficient to describe the deployment population.

E.1 SIMULATOR AND PRIOR SHIFT

One task is generated in three steps: draw its semantic path M, draw the coefficient vector β conditional on M, and then draw the observed input–response rows conditional on β. Specifically, draw K uniformly from $\{1, \dots, 8\}$ and draw $\mathcal{I}$ uniformly from the K-subsets of $\{1, \dots, 20\}$. Draw $U \sim \text{Unif}[0, 1]$. Under the training prior, $p_0(B = +1) = 0.8$. The semantic path $M = (K, \mathcal{I}, B, U)$ contains the task-level simulator choices exposed to T_1. The first experiment changes only B; the two-dimensional experiment below also allows the distribution of K to change. The coefficient vector β is instead a task-specific random effect drawn conditionally on M; its conditional distribution remains fixed between training and deployment. Thus, T_2 approximates the predictive distribution after integrating over the coefficient posterior,

$$T_2(dy \mid x, D, M) \approx \int p(dy \mid x, D, \beta, M)\, p(d\beta \mid D, M),$$

rather than conditioning on the realized β directly. We write $\mathcal{N}(\mu, v)$ for a normal distribution with mean μ and variance v. Conditional on M, the coefficient vector $\beta \in \mathbb{R}^{20}$ satisfies

$$\beta_j \sim \mathcal{N}\Big(B\sqrt{0.75/K},\, 0.25/K\Big) \quad (j \in \mathcal{I}), \qquad \beta_j = 0 \quad (j \notin \mathcal{I}).$$

For each row, draw $x_i \sim \mathcal{N}(0, I_{20})$ and

$$y_i = x_i^\top \beta + \epsilon_i, \qquad \epsilon_i \sim \mathcal{N}(0, 4\,16^{-U}).$$

Conditional on M, the active entries of β are independent. Conditional on (M, β), the input–response rows are independent. The number of discrete paths is $2\sum_{k=1}^{8}\binom{20}{k} = 527{,}898$; U adds one continuous coordinate. In the one-coordinate experiment, the deployment prior changes only $p(B = +1)$, from 0.8 to 0.2. Its prior-shift feature is $s(M) = B$.

In the notation of Section 2.2, the training distribution therefore factorizes as

$$p_0(dm)\, p(d\beta \mid m) \prod_{i=1}^{n} p(dx_i)\, p(dy_i \mid x_i, \beta).$$

Deployment replaces $p_0(dm)$ by a prior with a different marginal distribution for B. The coefficient law $p(d\beta \mid m)$, the input law, and the conditional response law remain unchanged. The controlled deployment distribution is therefore an allowed prior shift rather than a change in the conditional data-generating mechanism.

E.2 MODEL TRAINING

Under the training prior just defined, T_1 learns the posterior distribution of the semantic path given a data history, and T_2 learns the predictive distribution conditional on that path. Both networks instantiate the general architectures in Appendix C; after source training, their parameters remain fixed. To generate a semantic path with T_1, we use the autoregressive order

$$B \longrightarrow K \longrightarrow I_1 \longrightarrow \cdots \longrightarrow I_8 \longrightarrow U, \tag{67}$$

where $I_1 < \cdots < I_K$ are the ordered members of $\mathcal{I}$. Because $\mathcal{I}$ is an unordered set, it cannot be generated autoregressively without first choosing a unique ordering. We use its increasing representation

$$\mathcal{I} = \{I_1 < \cdots < I_K\}.$$

After T_1 draws $K = k$, only $I_1, \ldots, I_k$ are generated; the remaining index positions $I_{k+1}, \ldots, I_8$ contribute neither a conditional factor nor a training loss. The conditional distribution of the active indices factorizes as

$$T_1(\mathcal{I} \mid D, B, K = k) = \prod_{r=1}^{k} T_{1,I_r}\big(I_r \mid D, B, K = k, I_{1:r-1}\big). \tag{68}$$

At step r, T_1 conditions on the data history and the preceding semantic-path values and returns a categorical distribution for I_r. Categories that would repeat an index, violate increasing order, or make it impossible to complete a k-element subset are masked out. The remaining permitted indices are

$$\mathcal{S}_{I_r}(k, I_{1:r-1}) = \{j : I_{r-1} < j \leq 20 - (k - r)\}, \qquad I_0 = 0. \tag{69}$$

The lower bound requires the next index to exceed the preceding one. The upper bound reserves $k - r$ larger indices for the remaining steps. During training, the true preceding indices are supplied when evaluating each conditional factor. During deployment, an index is sampled from each conditional distribution and appended to the prefix before the next one is generated. The increasing order gives each set $\mathcal{I}$ one autoregressive representation; it does not make the order part of the semantic path.

Unlike T_1, T_2 does not generate the semantic path and therefore does not require its autoregressive representation. It receives the fixed-length numerical vector

$$c(m) = \big(\mathbf{1}\{1 \in \mathcal{I}\}, \ldots, \mathbf{1}\{20 \in \mathcal{I}\}, B, U\big) \in \mathbb{R}^{22}. \tag{70}$$

The first 20 entries indicate which input features are active. Their sum is

$$K = \sum_{j=1}^{20} \mathbf{1}\{j \in \mathcal{I}\},$$

so including K separately would repeat information already contained in $c(m)$.

The semantic posterior network T_1, the conditional predictive network T_2, and the direct PFN all represent tokens and hidden states in $\mathbb{R}^{192}$ and use eight attention heads. T_1 has four set-encoder layers and two autoregressive cross-attention layers, for 1,896,364 parameters. The conditional predictive T_2 has six layers and 1,810,084 parameters. Their combined parameter count is 3,706,448. The direct PFN has twelve layers and 3,735,844 parameters. Thus, the direct and Adaptive-PFN predictors differ in parameter count by less than 0.8%.

Training batches contain 128 tasks and alternate between data histories of length 8 and 32. Each task also supplies eight held-out query inputs and responses for the predictive training loss. We train T_1 for 8,192 updates on its semantic-path objective and T_2 for 16,384 updates on its conditional predictive objective. The two networks are trained on separate, independently generated streams of source tasks from the same training simulator. The direct PFN, which has approximately the same number of parameters, has only a predictive objective and is trained on the union of these two streams for 24,576 updates, so that its total source-task budget equals the combined budget used to train T_1 and T_2. All models use AdamW with peak learning rate 3×10^{-4}, weight decay 10^{-4}, gradient clipping at one, linear warm-up over the first $1/16$ of training, and cosine decay. For each model, we retain the checkpoint obtained after the stated number of updates; shifted performance is never used to select a checkpoint.

Both T_2 and the direct PFN use the Riemann predictive distribution in Appendix C.3 with $R = 100$ components. Before training any network, we draw 100,000 independent responses from the training simulator and use their empirical quantiles to fix the 101 borders

$$a_0 < a_1 < \cdots < a_{100}.$$

The borders are constructed once and shared by T_2, the direct PFN, and all training runs, so the predictors use the same response grid. We independently initialize and train three runs of T_1, T_2, and the direct PFN, labeled by seeds 0, 1, and 2. Their training master seeds are 93620912, 93650911, and 93650912, respectively; T_1, T_2, and the direct PFN are initialized independently within each run.

E.3 Prior estimation and adaptation

After source training, all network parameters and response borders remain fixed. We now instantiate the estimation and prediction steps of the pipeline. Because $s(M) = B$, an allowed deployment prior is completely determined by $p = p(B = +1)$. Under a scalar exponential tilt η, this probability is

$$p_\eta(B = +1) = \frac{0.8e^\eta}{0.8e^\eta + 0.2e^{-\eta}}. \tag{71}$$

Estimating the prior tilt is therefore equivalent to estimating the single deployment probability p.

The controlled deployment protocol separates prior estimation from predictive evaluation. First, 128 independently generated deployment data histories estimate the common value $p_*(B = +1)$. Second, 512 different deployment tasks evaluate predictions formed with that estimate. No response from a held-out query in the second stage is used to estimate the deployment prior.

The first stage compares the 128 deployment histories with a paired source sample. All deployment histories have length 32 and are generated with $p_*(B = +1) = 0.2$. We independently generate 512 histories conditional on $B = +1$ and pair each with the reflected history obtained through $(X, Y, B) \mapsto (X, -Y, -B)$. Evaluating T_1 on both members of every pair gives matched samples from the two sign-conditional distributions and reduces Monte Carlo variation. Let $r_t = T_1(B = +1 \mid D_t^*)$ for deployment history t. Because $B \in \{-1, +1\}$ and $s(M) = B$, the posterior semantic summary from Appendix B is

$$\bar{s}(D_t^*) = 2r_t - 1 = \mathbb{E}_{T_1}[B \mid D_t^*] = \mathbb{E}_{T_1}[s(M) \mid D_t^*].$$

Write $\bar{s}_+$ and $\bar{s}_-$ for the mean of this summary over the $B = +1$ and $B = -1$ members of the 512 pairs. Under a candidate deployment probability p, the estimated population mean is

$$\widehat{\psi}(p) = p\bar{s}_+ + (1 - p)\bar{s}_-.$$

The moment estimator chooses p so that $\widehat{\psi}(p)$ matches the mean of $\bar{s}(D_t^*)$ over the 128 population-fitting histories, subject to the prespecified tilt constraint $\eta \in [-4, 4]$. This is the stratified one-dimensional form of Equation 6.

The population-likelihood comparator instead uses the posterior sign probability r_t. Writing $p \in (0, 1)$ for a candidate value of $p(B = +1)$, it maximizes, over the same tilt constraint,

$$\mathcal{L}(p) = \sum_{t=1}^{128} \log\left\{ \frac{p}{0.8} r_t + \frac{1 - p}{0.2}(1 - r_t) \right\}.$$

The two terms inside the logarithm reweight the posterior probabilities of $B = +1$ and $B = -1$ by their candidate-to-training prior ratios. The known-shift comparator skips estimation and supplies $p = 0.2$ directly.

The fitted probability enters prediction only through these same prior ratios. For a new task with data history D, draw $M^{(1)}, \ldots, M^{(S)} \sim T_1(dm \mid D)$. Given an estimated deployment probability $\widehat{p}$, path j receives the unnormalized weight

$$\alpha_j(\widehat{p}) = \begin{cases} \widehat{p}/0.8, & B^{(j)} = +1, \\ (1 - \widehat{p})/0.2, & B^{(j)} = -1. \end{cases} \tag{72}$$

After normalizing these weights, the adapted prediction is

$$q_{\widehat{p}}(dy \mid x, D) \approx \sum_{j=1}^{S} \frac{\alpha_j(\widehat{p})}{\sum_{r=1}^{S} \alpha_r(\widehat{p})} T_2(dy \mid x, D, M^{(j)}). \tag{73}$$

Thus, the population histories determine $\widehat{p}$, $\widehat{p}$ changes the relative contribution of positive- and negative-effect paths, and T_2 converts the reweighted paths into a predictive distribution. The moment estimate, likelihood estimate, and known deployment probability all use this same prediction rule; neither T_1 nor T_2 is updated.

After estimating the deployment sign probability, we evaluate prediction on 512 new tasks that were used in neither the paired source sample nor the population fit. Each task has 32 held-out query pairs. We evaluate nested data histories of length 8 and 32 using the same queries, with 2,048 semantic-path samples per task. For each method, MSE is the mean squared difference between the held-out responses and their predictive means, and NLL is the mean negative log density assigned to the held-out responses by the mixed Riemann predictive distribution. We first average over the 32 queries within each task and then give every task equal weight.

We repeat the complete deployment experiment three times. Each repetition has a fresh paired source sample, 128 population-fitting histories, and 512 evaluation tasks; their master seeds are 93660910, 93660911, and 93660912. Each of the three trained model seeds is evaluated on every repetition. Reported means therefore weight the nine model-seed–deployment-repetition combinations equally. For each of 2,000 crossed bootstrap draws, we independently resample the three model seeds and three deployment repetitions with replacement, using master seed 93670910.

Table 1 reports the predictive results from the one-coordinate shift discussed in the main text.

Table 1: **One-coordinate controlled prior shift.** Lower MSE and NLL are better. Values average three independently trained model seeds crossed with three independent deployment repetitions, each with 512 evaluation tasks. The final column is the paired MSE reduction relative to the unadapted Adaptive-PFN under the same prior and at the same history length, with a crossed seed–deployment-repetition 95% interval. The population-likelihood fit is an empirical-Bayes comparator; "known shift" supplies the true deployment frequency rather than estimating it.

Data history	Population / predictor	MSE	NLL	MSE reduction
8	Training prior / direct PFN	2.1160	1.7814	–
8	Training prior / Adaptive-PFN, unadapted	2.1123	1.7818	reference
8	Training prior / moment adapted	2.1145	1.7822	-0.10% [-0.30, 0.00]
8	Shifted / direct PFN	2.2642	1.8163	–
8	Shifted / Adaptive-PFN, unadapted	2.2535	1.8143	reference
8	Shifted / moment adapted	2.1234	1.7845	5.77% [5.27, 6.16]
8	Shifted / likelihood adapted	2.1250	1.7847	5.70% [5.01, 6.18]
8	Shifted / known shift	2.1229	1.7843	5.79% [5.31, 6.19]
32	Shifted / direct PFN	1.8269	1.6456	–
32	Shifted / Adaptive-PFN, unadapted	1.7550	1.6299	reference
32	Shifted / moment adapted	1.7233	1.6235	1.81% [1.41, 2.17]
32	Shifted / likelihood adapted	1.7249	1.6239	1.72% [1.27, 2.14]
32	Shifted / known shift	1.7237	1.6236	1.79% [1.40, 2.17]

E.4 Two-dimensional prior shift

The preceding experiment uses the one-dimensional prior-shift feature $s(M) = B$. We next enlarge the allowed prior-shift family without changing or retraining either network. The prior-shift feature is now two-dimensional:

$$s(M) = \left(B, \frac{K - 4.5}{\sqrt{5.25}}\right).$$

This feature allows deployment to change both the effect-sign frequency and the distribution of the number of active input features. Its first coordinate is B; its second subtracts the training-prior mean of K and divides by its training-prior standard deviation, so that it has mean zero and variance one under that prior. The training distribution has $p_0(B = +1) = 0.8$ and uniform K on $\{1, \ldots, 8\}$.

The deployment distribution has $p_*(B = +1) = 0.2$ and exponentially tilts K so that $\mathbb{E}_*[K] = 2.5$. More precisely,

$$p_*(K = k) = \frac{p_0(K = k)\exp\{\eta_{*,2}(k - 4.5)/\sqrt{5.25}\}}{\sum_{r=1}^{8} p_0(K = r)\exp\{\eta_{*,2}(r - 4.5)/\sqrt{5.25}\}}, \qquad k = 1, \ldots, 8.$$

The known tilt is $\eta_* = (-1.386, -1.058)$. Writing $\rho_*(M) = p_*(M)/p_0(M)$, the population relative effective sample size is $\{\mathbb{E}_{p_0}[\rho_*(M)^2]\}^{-1} = 0.161$. Thus, prior reweighting has 16.1% of the effective sample size obtained under equal weights.

One population of 128 deployment data histories of length 32 and one independent set of 256 evaluation tasks are shared across the three trained seeds, so the predictions are compared on identical data. Each evaluation task has a data history of length 8, 32 held-out query pairs, and 2,048 posterior semantic-path samples. For moment matching, the deployment mean summary is matched to the exact training-prior moment map, which can be evaluated directly by summing over the two values of B and eight values of K. The population-likelihood comparator maximizes the approximate marginal likelihood in Equation 63 using the same T_1 posterior samples from the 128 population-fitting histories. Both fits are constrained to $\Theta = [-4, 4]^2$; the likelihood is optimized with L-BFGS. The population and evaluation tasks use master seed 93730916. Table 2 reports paired reductions relative to the unadapted Adaptive-PFN. All methods keep T_1 and T_2 fixed.

Table 2: **Two-dimensional controlled prior shift.** Each row uses an independently trained Adaptive-PFN. Estimated tilts correspond to effect sign and the standardized number of active input features. MSE reductions are relative to the same seed's unadapted Adaptive-PFN; higher reductions are better. ESS is the mean posterior-path effective sample size out of 2,048 for the likelihood fit.

Seed	$\widehat{\eta}_{\text{moment}}$	$\widehat{\eta}_{\text{likelihood}}$	MSE reduction			Likelihood ESS
			Moment	Likelihood	Known tilt	
0	$(-0.797, -0.226)$	$(-1.241, -1.223)$	5.83%	8.22%	8.32%	463
1	$(-0.802, -0.223)$	$(-1.267, -1.325)$	6.20%	8.53%	8.52%	423
2	$(-0.817, -0.241)$	$(-1.301, -1.617)$	6.21%	8.57%	8.54%	343
Mean	–	–	6.08%	8.44%	8.46%	409

The moment estimator has a mean posterior-path ESS of 1,315, but estimates the second tilt coordinate near -0.23, compared with its true value -1.058. The population-likelihood estimates produce lower posterior-path ESS, with a mean of 409, but their predictions nearly match the known-tilt prediction. Their seed-level 95% MSE-reduction intervals are $[5.78, 10.72]$, $[5.96, 11.12]$, and $[5.89, 11.27]$ percent. Moment matching also improves MSE in every seed, with intervals $[4.35, 7.23]$, $[4.78, 7.66]$, and $[4.69, 7.69]$ percent. These intervals use 2,000 task-level bootstrap resamples of the common 256 evaluation tasks, conditional on the trained seed and its fitted tilt. Mean NLL across seeds decreases from 1.8211 for the unadapted Adaptive-PFN to 1.7915 with moment matching and 1.7812 with population likelihood; both estimators reduce NLL for every seed.

The likelihood estimator performs better in this two-coordinate experiment, but not in the one-coordinate experiment. It relies on the full semantic-path posterior returned by T_1, whereas moment matching uses only the selected posterior summaries and has substantially greater posterior-path ESS. We use moment matching as the primary predictive estimator because it requires only these selected summaries; population likelihood provides a second comparison that uses the complete posterior returned by T_1.

E.5 Compatibility test

The preceding subsections estimate an allowed prior shift and use it for prediction. The compatibility test addresses a different question: whether any member of the allowed prior-shift family reproduces the selected properties of the observed data histories. In the controlled compatibility experiment, every allowed change alters only the frequency of B. We therefore construct a representation whose distribution is unchanged by $p(B = +1)$ but can change when the conditional relation between the inputs and response changes.

To make this scope concrete, we fix two excluded mechanisms before constructing the test. A source-only development pilot fixes their common severity at $\delta = 0.75$ before the validation populations

are generated; the validation does not tune this value. Let $j = \min \mathcal{I}$, let $\mu = x^\top \beta$, and let $c = \Phi^{-1}(0.75)$, where Φ is the standard-normal cumulative distribution function. The threshold-tree mechanism replaces the conditional mean by

$$\sqrt{1-\delta^2}\,\mu + \delta\|\beta\|\{2\mathbf{1}(|x_j| > c) - 1\},$$

while the heteroscedastic mechanism replaces ϵ by

$$\epsilon\,\frac{1+\delta\ \mathrm{erf}(x_j/\sqrt{2})}{\sqrt{1+\delta^2/3}}.$$

The first introduces an even, nonlinear change in the conditional mean; the second makes the conditional noise variance depend on an active input. Neither change can be produced by changing $p(B = +1)$. The threshold-tree term has unit variance and is uncorrelated with the linear conditional mean, so $\sqrt{1-\delta^2}$ preserves the total signal variance. Under the source input law, $\mathbb{E}[\mathrm{erf}(x_j/\sqrt{2})^2] = 1/3$, so the denominator in the heteroscedastic multiplier preserves its average squared value.

These mechanisms motivate a representation that retains general changes in the distribution of the input–response rows, together with information about nonlinear conditional means and input-dependent conditional variances. Its numerical parameters and normalizations are fixed using source tasks and prespecified random draws: no histories generated under either excluded mechanism are used to fit the representation or calibrate the null distribution. The same frozen representation, statistic, and rejection rule are used for every evaluated population, with no condition-specific tuning.

The test uses data histories of length 32 from populations of 128 tasks. The representation has three blocks. The set-valued block retains the empirical distribution of the observed input–response rows, while the location and scale blocks retain nonlinear mean dependence and input-dependent response variance, respectively.

We first define the set-valued block, which maps the unordered rows in one data history to a fixed-dimensional vector. Its first RFF level embeds the empirical distribution of rows within one history; its second level approximates a Gaussian kernel between the resulting history-level embeddings. Write $D = (X, Y) = \{(x_i, y_i)\}_{i=1}^n$ and $a_i = (x_i^\top, y_i)^\top \in \mathbb{R}^{21}$. Let $\widehat{\mu}_{\mathrm{row}}, \widehat{\sigma}_{\mathrm{row}} \in \mathbb{R}^{21}$ be the coordinatewise mean and standard deviation estimated from the 512 source construction tasks, and set

$$\widetilde{a}_i = (a_i - \widehat{\mu}_{\mathrm{row}}) \oslash \widehat{\sigma}_{\mathrm{row}},$$

where $\oslash$ denotes coordinatewise division. For $d = 1{,}024$, the first random Fourier feature (RFF) level approximates the Gaussian kernel $k_\ell(u, v) = \exp\{-\|u - v\|_2^2/(2\ell^2)\}$. Let ℓ_{row} be the median pairwise distance among standardized rows from the 512 source construction tasks. For $r = 1, \ldots, d$, independently draw

$$\omega_{\mathrm{row},r} \sim \mathcal{N}(0, \ell_{\mathrm{row}}^{-2} I_{21}), \qquad b_{\mathrm{row},r} \sim \mathrm{Unif}(0, 2\pi),$$

and collect the frequency vectors as the columns of $\Omega_{\mathrm{row}} \in \mathbb{R}^{21\times d}$ and the phases in $b_{\mathrm{row}} \in \mathbb{R}^d$. The resulting map is

$$\phi_{\mathrm{row}}(a_i) = \sqrt{\frac{2}{d}}\cos\big(\Omega_{\mathrm{row}}^\top \widetilde{a}_i + b_{\mathrm{row}}\big) \in \mathbb{R}^d,$$

where the cosine acts coordinatewise. Averaging over the observed rows removes their ordering:

$$\overline{\phi}(D) = \frac{1}{n}\sum_{i=1}^n \phi_{\mathrm{row}}(a_i).$$

Let $\widehat{\mu}_\phi, \widehat{\sigma}_\phi \in \mathbb{R}^d$ be the coordinatewise mean and standard deviation of $\overline{\phi}(D)$ across the same 512 source construction tasks, and write $v(D) = (\overline{\phi}(D) - \widehat{\mu}_\phi) \oslash \widehat{\sigma}_\phi$. Let ℓ_{task} be the median pairwise distance among the source values of $v(D)$. Independently for $r = 1, \ldots, d$, draw

$$\omega_{\mathrm{task},r} \sim \mathcal{N}(0, \ell_{\mathrm{task}}^{-2} I_d), \qquad b_{\mathrm{task},r} \sim \mathrm{Unif}(0, 2\pi),$$

and collect these vectors and phases in $\Omega_{\mathrm{task}} \in \mathbb{R}^{d\times d}$ and $b_{\mathrm{task}} \in \mathbb{R}^d$. The second RFF level is

$$\phi_{\mathrm{task}}(D) = \sqrt{\frac{2}{d}}\cos\big[\Omega_{\mathrm{task}}^\top v(D) + b_{\mathrm{task}}\big] \in \mathbb{R}^d.$$

All frequencies, phases, and bandwidths are drawn or estimated once from the source construction tasks and then frozen. To make this block invariant to the allowed response-sign change, let $D^- = (X, -Y)$ denote the reflected data history. We average the two histories before applying the frozen source normalization:

$$g_{\mathrm{set}}(D) = \frac{\frac{1}{2}\{\phi_{\mathrm{task}}(D) + \phi_{\mathrm{task}}(D^-)\} - \mu_{\mathrm{set}}}{\sigma_{\mathrm{set}}} \in \mathbb{R}^{1024}.$$

Here $\mu_{\mathrm{set}} \in \mathbb{R}^{1024}$ is the mean of the reflection-averaged vector over the 1,024 source normalization tasks and $\sigma_{\mathrm{set}} > 0$ is its root-mean-squared Euclidean deviation in the same sample.

The other two blocks target specific conditional changes. Let μ_x and σ_x be the scalar mean and standard deviation computed across every input entry in the 1,024 source normalization tasks. Within one history, define

$$z_{ij} = \frac{x_{ij} - \mu_x}{\sigma_x}, \qquad u_{ij} = \mathrm{erf}(z_{ij}/\sqrt{2}) = 2\Phi(z_{ij}) - 1, \qquad \widetilde{y}_i = \frac{y_i - \bar{y}}{\{n^{-1}\sum_{r=1}^{n}(y_r - \bar{y})^2\}^{1/2}}.$$

For each input coordinate j, set

$$e_{ij} = \frac{1}{2}(3u_{ij}^2 - 1), \qquad \bar{e}_j = \frac{1}{n}\sum_{i=1}^{n} e_{ij}, \qquad \bar{u}_j = \frac{1}{n}\sum_{i=1}^{n} u_{ij}.$$

The raw location and scale summaries are

$$L(D) = \sum_{j=1}^{20}\left\{\frac{1}{n}\sum_{i=1}^{n}(e_{ij} - \bar{e}_j)\widetilde{y}_i\right\}^2, \tag{74}$$

$$V(D) = \sum_{j=1}^{20}\left\{\frac{1}{n}\sum_{i=1}^{n}(u_{ij} - \bar{u}_j)(\widetilde{y}_i^2 - 1)\right\}^2. \tag{75}$$

erf is the error function. For a standard-normal input, u_{ij} is uniform on $[-1, 1]$, and $e_{ij} = \frac{1}{2}(3u_{ij}^2 - 1)$ is a centered even quadratic function of that input. Consequently, $L(D)$ measures squared dependence between the response and an even nonlinear function of each input. The centered odd term $u_{ij} - \bar{u}_j$ in $V(D)$ instead measures squared dependence between each input and the normalized squared response, and therefore targets input-dependent conditional variance. Let μ_L, σ_L and μ_V, σ_V be the respective means and standard deviations of these two summaries over the 1,024 source normalization tasks. The location and scale blocks are

$$g_{\mathrm{loc}}(D) = \frac{L(D) - \mu_L}{\sigma_L}, \qquad g_{\mathrm{scale}}(D) = \frac{V(D) - \mu_V}{\sigma_V}.$$

The full representation is

$$g(D) = \big(\sqrt{0.5}\, g_{\mathrm{set}}(D), 0.5\, g_{\mathrm{loc}}(D), 0.5\, g_{\mathrm{scale}}(D)\big).$$

The factors make the three blocks contribute weights 0.5, 0.25, and 0.25 to the linear kernel induced by g. These weights are fixed before validation: half of the kernel is assigned to the general set-valued block and the remaining half is divided equally between the location and scale blocks. The complete vector $g_{\mathrm{set}}(D)$ approximates a Gaussian-kernel representation of the empirical input–response distribution. The scalar $g_{\mathrm{loc}}(D)$ measures nonlinear changes in the conditional mean, while $g_{\mathrm{scale}}(D)$ measures input-dependent changes in the conditional variance.

Every quantity in g is fixed without deployment data. We use 512 source tasks to construct both RFF levels, 1,024 different source tasks to fit the normalizations and the two scalar blocks, and 4,096 further source tasks to fix the reference mean used by the test. These construction stages use master seed 93700910; the independent validation uses master seed 93710910.

This construction is invariant to the allowed shift. That shift changes only the frequency of the sign variable B. Under the source simulator, $(X, -Y)$ conditional on $B = +1$ has the same distribution as (X, Y) conditional on $B = -1$. The set-valued block averages these two histories explicitly;

$L(D)$ is unchanged because reversing Y changes the sign of each association before it is squared; and $V(D)$ depends on Y only through centered squares. Consequently,

$$g(D) = g(D^-).$$

The distribution of $g(D)$ is therefore the same for every allowed value of $p(B = +1)$. The null distribution of the test statistic does not depend on the unknown sign probability, so this experiment requires neither a test prior tilt nor a refitted parametric bootstrap.

Let $\widehat{\mu}_g$ be the mean of $g(D)$ over the 4,096 source tasks reserved for that purpose. For an observed population of $T = 128$ tasks, the statistic is

$$G_{\text{obs}} = T \left\| \frac{1}{T} \sum_{t=1}^{T} g(D_t^*) - \widehat{\mu}_g \right\|_2^2.$$

This is the linear-kernel MMD between the observed empirical distribution of $g(D)$ and the fixed source distribution, with the latter represented by its mean. We compute $B_{\text{boot}} = 99$ null statistics from independently simulated source populations of the same size and form the Monte Carlo value using the same plus-one upper-tail rank as Algorithm 1. This gives Monte Carlo values in increments of $1/(B_{\text{boot}} + 1) = 0.01$, including the nominal level 0.05. Conditional on the frozen feature map and $\widehat{\mu}_g$, the observed statistic and the null statistics are exchangeable under every allowed sign probability. We freeze the feature map and all sample sizes before validation. We then generate 200 independent base populations of 128 histories. Within each repetition, the allowed and excluded conditions are constructed from the same base population and compared with the same 99 simulated null populations; the 200 repetitions are independent.

Because the representation is exactly invariant to the sign-frequency shift, the null calibration applies equally to the training prior and to every allowed value of $p(B = +1)$. Table 3 displays the deployment value $p(B = +1) = 0.2$ and reports the resulting Type-I error and power estimates.

Table 3: **Calibration and power of the controlled compatibility test.** Rates are rejection counts divided by 200 independently generated populations; brackets give 95% Wilson intervals.

Data-generating condition	Rejects	Rate [95% CI]
Allowed prior tilt, $p(B = +1) = 0.2$ (null)	8/200	4.0% [2.0, 7.7]
Threshold-tree mechanism	200/200	100% [98.1, 100]
Heteroscedastic mechanism	200/200	100% [98.1, 100]

F London electricity experiment

F.1 Data and forecasting task

We use the public Low Carbon London smart-meter release and its dynamic time-of-use tariff schedule (Schofield et al., 2016; UK Power Networks, 2014). The data provide half-hourly electricity consumption for each participating household, while a separate calendar labels every half-hour as Normal, High, or Low tariff. We sum each pair of adjacent consumption readings to obtain hourly energy in kilowatt-hours (kWh). An hourly consumption value is observed only when both half-hour readings are observed, and an hourly tariff label is retained only when the two half-hours have the same label. We do not impute missing consumption readings, and candidate forecasting windows with a tariff change inside an hour are excluded.

We consider forecast origins at 05:00 local time so that every target covers the same 24-hour portion of the daily tariff schedule. The forecast origin is the first hour to be predicted. An origin is eligible when the preceding 48 hours are all under the Normal tariff, the following 24 hours include at least one High- or Low-tariff hour, and the resulting 72-hour window does not cross a daylight-saving offset change. For each household and eligible origin, one forecasting task uses the preceding 48 hours as its data history and the following 24 hours as its prediction target.

We index the hours in a task by $j \in \{-48, \ldots, -1, 0, \ldots, 23\}$, where $j = 0$ is the first target hour. Let y_j^{raw} be hourly energy in kWh and define $w_j = \log(1 + y_j^{\text{raw}})$. Using only the context index set

$\mathcal{C} = \{-48, \ldots, -1\}$, we compute the context mean $\mu_{\mathcal{C}}$ and context standard deviation $\sigma_{\mathcal{C}}$ as

$$\mu_{\mathcal{C}} = \frac{1}{48} \sum_{j \in \mathcal{C}} w_j, \qquad \sigma_{\mathcal{C}} = \max\left\{ 10^{-3}, \left[\frac{1}{48} \sum_{j \in \mathcal{C}} (w_j - \mu_{\mathcal{C}})^2 \right]^{1/2} \right\}.$$

The response supplied to every network is $\widetilde{y}_j = (w_j - \mu_{\mathcal{C}})/\sigma_{\mathcal{C}}$. Target responses are not used to calculate either normalization constant. For evaluation, we map a prediction $\widetilde{y}$ back to kWh as $\exp(\mu_{\mathcal{C}} + \sigma_{\mathcal{C}}\widetilde{y}) - 1$.

Each hour has the same eight-dimensional input format in simulated and London tasks. Let $h_0 \in \{0, \ldots, 167\}$ be the hour of the local calendar week at $j = 0$, with Monday 00:00 represented by zero. The input at hour j is

$$x_j = \left(\frac{j}{336}, \sin \frac{2\pi(h_0 + j)}{24}, \cos \frac{2\pi(h_0 + j)}{24}, \sin \frac{2\pi(h_0 + j)}{168}, \cos \frac{2\pi(h_0 + j)}{168}, H_j, L_j, \phi(\sigma_{\mathcal{C}}) \right).$$

The first coordinate is relative time measured from the forecast origin and scaled by the fixed 336-hour reference-history length used by the simulator. Thus, the short London context uses values $-48/336, \ldots, -1/336$, while target values run from zero to $23/336$. The next two coordinates locate an hour within its 24-hour daily cycle, and the following two locate it within its 168-hour weekly cycle.

The indicator H_j equals one during a High-tariff hour and zero otherwise; L_j is defined analogously for Low-tariff hours. They cannot both equal one, and $(H_j, L_j) = (0, 0)$ denotes a Normal-tariff hour. The tariff schedule is known for both context and target hours when the forecast is issued. Finally, the response-scale feature is constant within a task and equals

$$\phi(\sigma) = \frac{\log \sigma - \frac{1}{2}(\log 0.03 + \log 1.00)}{\frac{1}{2}(\log 1.00 - \log 0.03)}.$$

Here $\sigma > 0$ is the generic argument of the encoding; each task uses $\sigma = \sigma_{\mathcal{C}}$. Thus, $\phi(\sigma_{\mathcal{C}})$ records how much variation was removed when the task responses were normalized. Supplying it as an input allows the network to relate predictions on the normalized scale to the original $\log(1 + y)$ scale.

Before examining whether the required readings are observed, we fix three disjoint groups of households assigned to the dynamic tariff: 256 for population fitting, 128 for development, and 256 for evaluation. This assignment precedes all checks of whether the readings required by the subsequent analyses are observed. Of the 256 population-fitting households, 248 have enough observed rows for the tariff-response regression below to have full column rank. Their 336-hour data histories contain between 258 and 335 observed hours, with median 335. These longer histories are used only to estimate the deployment prior and assess compatibility; prediction still uses the 48-hour data histories defined above. The development period contains 17 fixed eligible forecast origins from April through June 2013. The evaluation period contains 27 fixed eligible origins from July through December 2013; 25 yield at least one household task with all readings observed throughout the 72-hour window containing its data history and prediction target. Evaluation outcomes are not used until the population fit and compatibility decision have been fixed.

F.2 Synthetic training simulator

Every source task has the same hourly inputs and 24-hour prediction target as a London task, but its response is generated entirely by the synthetic training simulator. Tasks alternate equally between two data-history designs. The *short view* uses the final 48 hours of a 336-hour reference history and assigns every one of those hours the Normal tariff, matching the London prediction task. The *exposed view* uses the complete 336-hour reference history and independently assigns High, Low, and Normal tariffs with probabilities 0.1, 0.1, and 0.8 at each hour. In both designs, the 24-hour target contains one contiguous High- or Low-tariff block. Its tariff is chosen with equal probability, its length is uniform on $\{3, 6, 12\}$ hours, and its starting hour is uniform among the positions at which the block fits; all remaining target hours are Normal tariff. This produces one contiguous non-Normal-tariff block while varying its tariff, duration, and position. The probabilities in the exposed view repeatedly show T_1 both non-Normal tariffs while leaving most reference hours Normal; the three target lengths

occupy one eighth, one quarter, or one half of the 24-hour horizon. The forecast origin is uniform over the 168 hours of the week. The short view reproduces the setting in which the immediate data history contains no direct evidence about the tariff associations; the exposed view teaches T_1 to recover those associations when such evidence is present.

After drawing the context and target tariff schedules, the simulator draws the semantic path

$$M = (C_{\text{trend}}, C_{\text{season}}, C_{\text{change}}, C_{\text{noise}}, U_\rho, U_{\text{snr}}, \beta_H, \beta_L).$$

The eight coordinates and their independent training-prior distributions are listed in Table 4. The first four specify whether the series contains a trend, which seasonal components it contains, whether it contains a level change, and whether its additive observation-noise terms are Gaussian or scaled Student-t_5. The variables U_ρ and U_{snr} determine the autoregressive persistence and signal-to-noise ratio through the mappings in the table. The variables β_H and β_L determine how the response differs during High- and Low-tariff hours.

Table 4: **Synthetic training prior for the London experiment.** All eight semantic coordinates are sampled independently for every task. Here, $\text{TN}(\mu, \sigma^2; [a, b])$ denotes a normal distribution with mean parameter μ and variance parameter σ^2, truncated to $[a, b]$.

Coordinate	Training-prior distribution	Meaning
C_{trend}	uniform on {absent, linear}	trend component
C_{season}	uniform on {absent, daily, daily+weekly}	seasonal components
C_{change}	uniform on {absent, present}	permanent level shift
C_{noise}	uniform on {Gaussian, Student-t_5}	observation-noise distribution
U_ρ	$\text{Unif}[0, 1]$	AR(1) lag-one correlation $\rho = 0.95U_\rho$
U_{snr}	$\text{Unif}[0, 1]$	signal-to-noise variance ratio $0.25\,16^{U_{\text{snr}}}$
β_H	$\text{TN}(0, 0.25^2; [-0.75, 0.75])$	High-tariff association
β_L	$\text{TN}(0, 0.25^2; [-0.75, 0.75])$	Low-tariff association

Conditional on M, the simulator generates hours $t = 0, \ldots, 359$. Hours $0, \ldots, 335$ form the two-week reference history and hours $336, \ldots, 359$ form the 24-hour prediction target. We now specify the effect of each coordinate of M in the order in which it appears.

Trend, C_{trend}. If $C_{\text{trend}} = \text{linear}$, the latent signal includes the linear component

$$b_{\text{trend},t} = \frac{t - 336}{336};$$

if $C_{\text{trend}} = \text{absent}$, this component is omitted.

Seasonality, C_{season}. For a period $P \in \{24, 168\}$, draw independent task-specific coefficients $A_{P,k}, B_{P,k} \sim \mathcal{N}(0, k^{-2})$ for $k = 1, 2, 3$ and define

$$b_{P,t} = \sum_{k=1}^{3} \left[A_{P,k} \sin\left\{ \frac{2\pi k(h_0 + t - 336)}{P} \right\} + B_{P,k} \cos\left\{ \frac{2\pi k(h_0 + t - 336)}{P} \right\} \right].$$

The value absent omits both seasonal components, daily includes only $b_{24,t}$, and $\text{daily} + \text{weekly}$ includes both $b_{24,t}$ and $b_{168,t}$. Because the coefficient variance decreases as k^{-2}, higher harmonics have smaller expected squared amplitudes. Their realized coefficients remain outside the semantic path.

Level change, C_{change}. If $C_{\text{change}} = \text{present}$, draw τ_Δ uniformly from $\{72, \ldots, 288\}$ and include

$$b_{\text{change},t} = \mathbf{1}\{t \geq \tau_\Delta\}.$$

This is a permanent shift rather than a temporary spike, and the range of τ_Δ leaves reference observations on both sides of the change. If $C_{\text{change}} = \text{absent}$, the component is omitted.

Observation noise, C_{noise}. This coordinate selects the distribution of mutually independent, unit-variance noise terms. If $C_{\text{noise}} = \text{Gaussian}$, draw $\epsilon_t \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1)$. If $C_{\text{noise}} = \text{Student-}t_5$, draw $T_t \overset{\text{i.i.d.}}{\sim} t_5$ and set

$$\epsilon_t = \sqrt{3/5}\, T_t.$$

Because a Student-t_5 variable has variance $5/3$, this scaling gives ϵ_t unit variance.

Persistence, U_ρ. The mapping $\rho = 0.95U_\rho$ determines a stationary Gaussian autoregressive process of order one, abbreviated AR(1):

$$\begin{aligned} v_0 &\sim \mathcal{N}(0,1), \\ \xi_t &\overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,1), \\ v_t &= \rho v_{t-1} + \sqrt{1-\rho^2}\,\xi_t, \qquad t = 1, \ldots, 359. \end{aligned}$$

The initial value v_0 is independent of the variables ξ_t. The AR(1) component v_t is always included. Its correlation at lag one is ρ, so a larger U_ρ produces a more slowly varying component.

Signal-to-noise ratio, U_{snr}. This coordinate sets

$$\text{SNR} = 0.25\,16^{U_{\text{snr}}} \in [0.25, 4].$$

After the latent signal is constructed on unit scale, the observation noise is multiplied by $\text{SNR}^{-1/2}$. Hence SNR is the ratio of signal to observation-noise variance: values above one favor the signal, while values below one favor the noise.

High-tariff association, β_H. The term $\beta_H H_t$ is added to the unnormalized log response. It is zero during Normal- and Low-tariff hours and equals β_H during High-tariff hours.

Low-tariff association, β_L. Analogously, $\beta_L L_t$ is zero during Normal- and High-tariff hours and equals β_L during Low-tariff hours. These two coordinates describe predictive associations and are not interpreted as causal effects.

It remains to combine the structural components. For the always-present AR(1) component and each trend, seasonal, or level-change component selected above, use the reference history to define

$$\bar{b} = \frac{1}{336}\sum_{t=0}^{335} b_t, \qquad s_b^2 = \frac{1}{336}\sum_{t=0}^{335}(b_t - \bar{b})^2, \qquad \widehat{b}_t = \frac{b_t - \bar{b}}{s_b}.$$

Draw an independent sign d_b uniformly from $\{-1, +1\}$ and magnitude $m_b \sim \text{Unif}[0.5, 1.5]$, and set

$$\theta_b = \frac{d_b m_b}{\{\sum_h m_h^2\}^{1/2}},$$

where the sum runs over these same components. We then define

$$\widetilde{f}_t = \sum_b \theta_b \widehat{b}_t, \qquad \bar{f} = \frac{1}{336}\sum_{t=0}^{335}\widetilde{f}_t, \qquad s_f^2 = \frac{1}{336}\sum_{t=0}^{335}(\widetilde{f}_t - \bar{f})^2,$$

and set $f_t = (\widetilde{f}_t - \bar{f})/s_f$. Normalizing both the components and their sum prevents paths with more active components from having systematically larger signals.

The simulator combines this signal with the observation-noise terms selected by C_{noise}:

$$\widetilde{z}_t = f_t + \text{SNR}^{-1/2}\epsilon_t.$$

Using only the reference history, define

$$\bar{z}_{\text{ref}} = \frac{1}{336}\sum_{t=0}^{335}\widetilde{z}_t, \qquad s_{\text{ref}}^2 = \frac{1}{336}\sum_{t=0}^{335}(\widetilde{z}_t - \bar{z}_{\text{ref}})^2.$$

We then set

$$z_t = \frac{\widetilde{z}_t - \bar{z}_{\text{ref}}}{s_{\text{ref}}}$$

for all 360 hours, so the prediction target does not affect either normalization constant.

Finally, draw a task-specific background-series scale a according to $\log a \sim \text{Unif}[\log 0.03, \log 1]$ and form the response on the unnormalized $\log(1+y)$ scale,

$$r_t = a z_t + \beta_H H_t + \beta_L L_t.$$

The harmonic coefficients, change location, component signs and magnitudes, background-series scale, AR(1) draws, and observation-noise draws are sampled afresh for each task. They vary across tasks sharing the same semantic path and are not coordinates of M. The short view uses $\mathcal{C} = \{288, \dots, 335\}$, and the exposed view uses $\mathcal{C} = \{0, \dots, 335\}$. For either context index set, define

$$\mu_{\mathcal{C}} = \frac{1}{|\mathcal{C}|} \sum_{t \in \mathcal{C}} r_t, \qquad \sigma_{\mathcal{C}}^2 = \frac{1}{|\mathcal{C}|} \sum_{t \in \mathcal{C}} (r_t - \mu_{\mathcal{C}})^2.$$

The network response is $(r_t - \mu_{\mathcal{C}})/\sigma_{\mathcal{C}}$ for both context and target hours. Therefore, target values never enter normalization. The model also receives the same deterministic encoding $\phi(\sigma_{\mathcal{C}})$ used for the London tasks. This input records the context variation by which the tariff associations are divided during response normalization.

F.3 Model training

The semantic posterior network T_1 generates all eight coordinates of M in the order in which they are listed above; every coordinate is active. The numerical representation supplied to T_2 is

$$\begin{aligned} c(m) = \big(&\mathbf{21}\{C_{\text{trend}} = \text{linear}\} - 1, \mathbf{21}\{C_{\text{season}} \neq \text{absent}\} - 1, \\ &\mathbf{21}\{C_{\text{season}} = \text{daily + weekly}\} - 1, \mathbf{21}\{C_{\text{change}} = \text{present}\} - 1, \\ &\mathbf{21}\{C_{\text{noise}} = \text{Student-}t_5\} - 1, 2U_\rho - 1, 2U_{\text{snr}} - 1, \beta_H/0.75, \beta_L/0.75\big) \in \mathbb{R}^9. \end{aligned} \tag{76}$$

The two seasonality indicators distinguish absent, daily, and daily-plus-weekly seasonality, so this representation is injective in M. T_2 returns one predictive distribution for each of the 24 normalized target responses. The direct PFN receives the observations but not M. All three models see tasks from the same training prior. The training streams for T_1 and T_2 are independent, while the direct PFN replays their union so that its task budget equals the combined Adaptive-PFN budget.

All networks have width 192 and eight attention heads. The Adaptive-PFN contains 3,636,974 parameters: 1,831,690 in T_1 and 1,805,284 in T_2. The direct PFN contains 3,636,520 parameters. We train T_1 and T_2 for 4,096 updates each and the direct PFN for 8,192 updates, with batches of 128 tasks. Optimization uses AdamW, peak learning rate 3×10^{-4}, weight decay 10^{-4}, gradient clipping at one, linear warm-up over $1/16$ of training, and cosine decay. The common Riemann output grid has 100 buckets, estimated from 100,000 source responses, with half-normal tails. Seeds 0, 1, and 2 independently train the same fixed-budget networks and share this output grid. We use each seed's final checkpoint; no London observation updates its weights.

F.4 Deployment prior estimation

Evidence of prior shift. We first check whether the London tariff associations differ from their training prior without using T_1. For each of the 248 population-fitting households, we regress observed $\log(1 + \text{consumption})$ on High and Low tariff indicators, hour-of-day and day-of-week indicators, and a linear trend. Appendix F.5 gives the complete regression design. The fitted High and Low coefficients have means -0.0418 and 0.0062. Their raw standard deviations contain both between-household variation and uncertainty in the fitted coefficients. For tariff $r \in \{H, L\}$, let S_r^2 be the sample variance of the 248 fitted coefficients $\widehat{\theta}_{i,r}$, and let $\widehat{V}_{i,r}$ be the corresponding estimated coefficient variance. We use

$$\widehat{\tau}_r = \left[\max \left\{ S_r^2 - \frac{1}{248} \sum_{i=1}^{248} \widehat{V}_{i,r}, 0 \right\} \right]^{1/2}$$

as the estimated between-household standard deviation. The resulting standard deviations are 0.0945 (95% household-bootstrap interval $[0.0737, 0.1141]$) for High and 0.0485 ($[0.0411, 0.0554]$) for Low. By comparison, the truncated training prior has standard deviation 0.2466 for each coefficient. The intervals use 2,000 bootstrap samples of the 248 households and recompute the variance correction in every sample.

This comparison uses only the population-fitting data histories. It shows that the London tariff associations are substantially more concentrated than under the training prior and identifies population information that the frozen predictor can use after deployment.

Output calibration. The calibrated direct PFN tests whether the fitted High- and Low-tariff means and between-household variances are enough to adapt the final output distribution through a simple affine correction. After undoing the context normalization, let L denote its predictive distribution on the $\log(1+y^{\text{raw}})$ scale, and let Q_α be the quantile of L at probability α. We summarize its center and scale by

$$c = Q_{0.5}, \qquad \widehat{\sigma} = \frac{Q_{0.9} - Q_{0.1}}{2\Phi^{-1}(0.9)} = \frac{Q_{0.9} - Q_{0.1}}{2(1.28155)},$$

where Φ is the standard-normal cumulative distribution function. Thus, $\widehat{\sigma}$ is the Gaussian-equivalent scale of the central 80% predictive interval; it is not estimated from observed target responses. This scale is needed to replace the training-prior contribution to predictive variance. Using central rather than extreme quantiles keeps that replacement from depending strongly on the fitted half-normal tails of the Riemann predictive distribution.

For tariff state $r \in \{H, L\}$, let μ_r and τ_r^2 be the population-fitting mean and corrected between-household variance above, and let σ_0^2 be the corresponding training-prior variance. We replace L on active hours by

$$L_{\text{cal}} = c + \mu_r + a_r(L - c), \qquad a_r^2 = \frac{\max\{\widehat{\sigma}^2 - \sigma_0^2 + \tau_r^2, 10^{-8}\}}{\widehat{\sigma}^2}.$$

This preserves the shape of the direct PFN distribution while replacing its tariff-response mean and variance. Normal-tariff hours are unchanged. The variance floor is never active in the held-out evaluation. Quantiles use the affine transformation above, and NLL uses its exact change of variables. The correction is fixed from population-fitting data histories, applied without neural retraining, and evaluated on the same evaluation cases as Adaptive-PFN.

Prior estimation. Define $Z_H = \beta_H/0.25$ and $Z_L = \beta_L/0.25$. Let v_0 be the variance of a standard normal variable truncated to $[-3, 3]$. Under the training prior, Z_H and Z_L are independent draws from this symmetric distribution, so their means are zero and their common second moment is v_0. The four population tilt statistics are

$$s(M) = (Z_H, Z_L, Z_H^2 - v_0, Z_L^2 - v_0).$$

To see how they change the prior, write the tilt as $\eta = (\eta_H, \eta_L, \kappa_H, \kappa_L)$. If p_0^Z denotes the training-prior density of either standardized association, then, for $(z_H, z_L) \in [-3, 3]^2$, the tilted joint density is

$$\begin{aligned} p_\eta^{HL}(z_H, z_L) \propto\ & p_0^Z(z_H) p_0^Z(z_L) \exp\{\eta_H z_H + \eta_L z_L\} \\ & \times \exp\{\kappa_H(z_H^2 - v_0) + \kappa_L(z_L^2 - v_0)\}. \end{aligned}$$

For association $r \in \{H, L\}$, let (η_r, κ_r) denote its linear and quadratic tilt coordinates. Its marginal density satisfies

$$p_{\eta_r,\kappa_r}^Z(z) \propto p_0^Z(z) \exp\{\eta_r z + \kappa_r(z^2 - v_0)\}.$$

Because p_0^Z is symmetric, for every $z > 0$, $p_{\eta_r,\kappa_r}^Z(z)/p_{\eta_r,\kappa_r}^Z(-z) = \exp(2\eta_r z)$. Thus, $\eta_r > 0$ favors positive associations and $\eta_r < 0$ favors negative associations. The quadratic factor $\exp\{\kappa_r(z^2 - v_0)\}$ instead assigns the same weight to z and $-z$. When $\kappa_r > 0$, its weight increases with $|z|$, favoring associations farther from zero; when $\kappa_r < 0$, it favors associations closer to zero. Subtracting v_0 makes the quadratic statistic have training-prior mean zero but does not change any relative weight because $\exp(-\kappa_r v_0)$ is constant in z. The absence of a $Z_H Z_L$ term means that the tilt changes the two marginal distributions separately and preserves their independence.

For each seed and population-fitting data history, 256 paths from T_1 approximate the posterior mean $\bar{s}(D)$ in Section 2.4. Applying the fixed whitening transformation gives $z(D)$, and averaging $z(D)$ over the 248 histories gives the deployment target for that seed's moment estimator. Whitening uses 2,048 independent source tasks and fitting uses a separate bank of 4,096 source tasks. Every bank uses the observed 336-hour London tariff schedule, cycles through the 248 observed-hour masks, and matches each household's regression-residual root mean square. The latter is the square root of the mean squared residual over that household's observed hours in the regression given in Appendix F.5. Thus, the source tasks used for the fit reproduce the population-fitting tariff schedule, missingness, and residual scales rather than drawing a new tariff schedule from the training simulator. We optimize the finite-bank objective in Equation 22 over $[-80, 80]^4$ using the deterministic multistart L-BFGS

procedure specified in Appendix B.2.3. The optimizer and parameter box are identical across seeds. Table 5 reports the fitted tilts and overlap.

Seed zero also underwent an independent 4,096-task bank check. Its primary and check-bank fits were $(-0.779, -2.902, -3.835, -29.244)$ and $(-0.996, -3.425, -3.830, -24.257)$, with source-bank ESS 339.5 and 392.2. Here the Jacobian is the derivative, with respect to the tilt, of the reweighted source mean in Equation 22. Its smallest singular values were 4.25×10^{-4} and 6.41×10^{-4}, respectively. These small values show weak identification in at least one direction, while the change between the two fits occurs mainly in the fourth coordinate. Table 10 checks the predictive consequence of changing the path count.

Table 5: **London population fits across training seeds.** Each seed uses its own T_1 and the same 248 population-fitting households, whitening rule, 4,096-task source fit bank, and box-bounded moment objective. The objective is the squared discrepancy between the reweighted source mean and the population mean of the standardized semantic summaries. Bank ESS and path ESS refer, respectively, to the source-bank and posterior-path effective sample sizes defined in Equations 25 and 26. Bank ESS is out of 4,096 source tasks. The path ESS is out of 256 paths, and its mean and minimum are taken over the evaluation household–event pairs.

Seed	Fitted $\widehat{\eta}$	Objective	Bank ESS	Mean path ESS	Min. path ESS
0	$(-0.779, -2.902, -3.835, -29.244)$	$4.63 \cdot 10^{-16}$	339.5	22.15	10.57
1	$(1.011, -10.170, -6.317, -80.000)$	$4.75 \cdot 10^{-4}$	164.5	11.84	3.99
2	$(1.181, -0.598, -4.957, -9.900)$	$5.12 \cdot 10^{-15}$	529.9	32.18	16.29

Seed 1 has the weakest fourth-coordinate identification: its reported fit reaches the lower parameter-box boundary. In the 200 household-bootstrap refits, 174 reach that boundary and 132 meet the projected-gradient tolerance of 10^{-6}. The mean objective is 4.77×10^{-4}, the mean projected-gradient norm is 9.10×10^{-7}, and the central 95% range of source-bank ESS fractions is $[0.032, 0.049]$. Table 8 propagates these refits to prediction; its seed-1 intervals remain entirely positive.

F.5 Compatibility test

The compatibility representation is computed directly from each data history and does not use T_1. To distinguish the histories used here from the 48-hour prediction histories, write D_i^{fit} for the 336-hour population-fitting history of household i, and let $\mathcal{O}_i \subseteq \{0, \dots, 335\}$ contain its observed hours. For each $t \in \mathcal{O}_i$, define

$$r_{it} = \log(1 + y_{it}^{\mathrm{raw}})$$

and the regression row

$$q_t = \left(1, \mathbf{1}\{h_t = 1\}, \dots, \mathbf{1}\{h_t = 23\}, \mathbf{1}\{d_t = 1\}, \dots, \mathbf{1}\{d_t = 6\}, \frac{2t}{335} - 1, H_t, L_t\right)^\top.$$

Here $h_t \in \{0, \dots, 23\}$ is the local hour of day, $d_t \in \{0, \dots, 6\}$ is the local day of the week, and H_t and L_t are the High- and Low-tariff indicators defined in Appendix F.1. Hour zero and weekday zero are the reference categories. The time coordinate ranges from -1 to 1 over the data history.

Let Q_i be the matrix with rows $q_t^\top$, $t \in \mathcal{O}_i$, and let $r_i = (r_{it})_{t\in\mathcal{O}_i}$. When Q_i has full column rank, we compute the ordinary least-squares estimate

$$\widehat{\theta}_i = (Q_i^\top Q_i)^{-1} Q_i^\top r_i.$$

The hour-of-day and weekday coefficients account for recurring calendar variation, while the time coefficient accounts for linear drift across the history. The final two coefficients measure the remaining additive associations of High and Low tariff hours with $\log(1 + y^{\mathrm{raw}})$. We define

$$g(D_i^{\mathrm{fit}}) = \begin{pmatrix} \widehat{\theta}_{i,H} \\ \widehat{\theta}_{i,L} \end{pmatrix} \in \mathbb{R}^2.$$

A household is retained only when Q_i has full column rank, so both coordinates are uniquely defined. The first coordinate compares consumption during High-tariff and Normal-tariff hours after

accounting for hour of day, weekday, and linear drift; the second makes the analogous comparison for Low-tariff hours. These are observable associations rather than causal effects.

We prespecified this representation before examining the deployment population and without knowing how that population might depart from the simulator. Its choice follows from the scientific scope of the application: the allowed tilt changes the training-prior distribution of the High- and Low-tariff associations, so the compatibility test should examine whether the fitted prior-shift model reproduces their observable counterparts in real data. The test compares the joint distribution of the two coordinates, rather than only their means, and can therefore reveal differences in their variability or dependence across households. The balanced common coefficient component considered below is one particular departure used to evaluate this sensitivity; it did not determine the construction of g.

Source reference tasks reproduce the London tariff schedule, observed-data masks, and regression-residual root mean squares, and g is computed from each source task by the same regression.

Applying g to the 248 population-fitting histories gives 248 points in $\mathbb{R}^2$, one pair of estimated tariff associations per household. To determine whether these points are compatible with an allowed prior shift, we compare their distribution with the distribution of g produced by source tasks after reweighting those tasks toward the fitted prior tilt.

Before making this comparison, we place the two coordinates of g on comparable scales. From a separate construction sample of 1,024 source tasks, let $\mu_g \in \mathbb{R}^2$ and $\sigma_g \in \mathbb{R}^2$ be the coordinatewise mean and standard deviation. For any observed or simulated 336-hour history D^{fit}, we use

$$g_{\mathrm{std}}(D^{\mathrm{fit}}) = \frac{g(D^{\mathrm{fit}}) - \mu_g}{\sigma_g},$$

where division is coordinatewise. This construction sample is used only to fix the transformation and is independent of both the observed population and the source reference bank used by the test.

We then measure the similarity between two standardized representations with the Gaussian RBF kernel

$$k_\ell(u, v) = \exp\left(-\frac{\|u - v\|_2^2}{2\ell^2}\right). \tag{77}$$

This kernel assigns a value close to one to households with similar pairs of tariff associations and a value approaching zero as the pairs move farther apart. Its bandwidth controls how quickly this similarity decreases with Euclidean distance. We set it to the median positive distance between pairs in the standardized construction sample and freeze the resulting value, 1.676, before applying the test.

The remaining steps instantiate the procedure in Appendix D. The test fit does not use T_1. Define the observable moment vector

$$u(D^{\mathrm{fit}}) = \left(\frac{\widehat{\theta}_H(D^{\mathrm{fit}})}{0.25}, \frac{\widehat{\theta}_L(D^{\mathrm{fit}})}{0.25}, \left\{\frac{\widehat{\theta}_H(D^{\mathrm{fit}})}{0.25}\right\}^2, \left\{\frac{\widehat{\theta}_L(D^{\mathrm{fit}})}{0.25}\right\}^2\right).$$

We whiten $u(D^{\mathrm{fit}})$ using the same independent 1,024-task construction sample and fit the test prior tilt $\widetilde{\eta}$ by matching its mean over the 248 population-fitting histories to its reweighted mean in a separate bank of 4,096 source tasks. These four moments correspond to the same linear and quadratic coordinates used by the allowed prior tilt, with centering handled by the source whitening. We optimize over $[-80, 80]^4$ by deterministic multistart L-BFGS, allowing 100 iterations per start and requiring a projected-gradient norm of at most 10^{-6}. The fitted tilt then assigns importance weights to a different reference bank of 2,048 source tasks. This weighted bank represents the distribution of g under the fitted member of the allowed prior-shift family. We compare it with the 248 observed representations using the biased empirical MMD in Equation 66. All pairwise kernel values are evaluated directly, without a random-feature approximation, and the resulting squared MMD is multiplied by 248.

Two further independent banks, each containing 4,096 source tasks, supply the simulated populations used to validate and calibrate the test. The outer bank supplies the populations used to estimate Type-I error and power. For each of the 99 parametric-bootstrap replicates, we sample 248 tasks with replacement from the bootstrap bank using the fitted tilt weights, refit $\widetilde{\eta}$, and recompute the complete MMD statistic, following Algorithm 1. Thus, the construction, tilt fit, MMD reference, outer validation, and bootstrap samples are mutually independent source banks.

Before applying the test to the London population, we evaluate its finite-sample behavior at the same population size and significance level. The Type-I error is the probability that the test rejects even though the population was generated by a member of the allowed prior-shift family. We estimate it under two null distributions: the training prior, corresponding to $\eta = 0$, and the test tilt fitted to the London population,

$$\widetilde{\eta}_{\text{London}} = (-1.490, 3.826, -5.738, -80.000).$$

For each null distribution, we independently sample 60 populations of 248 representations from the reweighted outer bank and apply the complete test. At the nominal level $\alpha = 0.05$, a well-calibrated test should reject approximately 5% of these populations.

Power is instead the probability that the test rejects a specified population outside the allowed family. We evaluate power at the level of the selected representation g. For each coefficient pair V_{null} sampled under $\widetilde{\eta}_{\text{London}}$ from the outer bank, define

$$V_{\text{alt}} = V_{\text{null}} + \begin{pmatrix} C \\ C \end{pmatrix}, \qquad C \in \{-0.25, +0.25\}.$$

Within each population, exactly half of the households receive each value of C, with the assignments randomly permuted. The added term therefore leaves the population mean of each coefficient unchanged, but makes the High- and Low-tariff coefficients move together across households. The allowed tilt can change the marginal means and concentrations of the two simulated tariff associations, but it preserves their independence and therefore cannot reproduce this additional dependence. We construct 60 independently sampled populations of coefficient pairs under this excluded mechanism; the fraction rejected estimates the power against this particular departure.

After this finite-sample calibration check, we apply the procedure to the observed London population. Its fitted test tilt reaches the lower boundary of the parameter box in the fourth coordinate, so every bootstrap replicate uses the same constrained fitting rule. The observed statistic is $G_{\text{obs}} = 0.1218$. All 99 fitted-null bootstrap statistics are at least this large, giving

$$p_{\text{MC}} = \frac{1 + 99}{99 + 1} = 1.00.$$

The test therefore does not reject compatibility at level 0.05. Table 6 reports the Type-I error and power validation.

Table 6: **Calibration and power of the London compatibility test.** Type-I error is rejection when the null is true; power is rejection under the specified excluded mechanism. Rates are rejection counts divided by 60 independently sampled populations; brackets give 95% Wilson intervals.

Data-generating condition	Quantity	Rejects	Estimate [95% CI]
Training prior, $\eta = 0$ (null true)	Type-I error	5/60	8.3% [3.6, 18.1]
Fitted London test tilt (null true)	Type-I error	0/60	0.0% [0.0, 6.0]
Fitted tilt plus balanced common coefficient component (null false)	Power	60/60	100% [94.0, 100]

F.6 Metrics and results

For a household–event pair, let S be the evaluated set of forecast hours, y_h the observed consumption, and $Q_{q,h}$ the predicted quantile at $q \in \mathcal{Q} = \{0.1, 0.2, \ldots, 0.9\}$. With pinball loss $\rho_q(u) = u\{q - \mathbf{1}(u < 0)\}$, we compute

$$\text{WQL}_S = \frac{1}{9} \sum_{q \in \mathcal{Q}} \frac{2 \sum_{h \in S} \rho_q(y_h - Q_{q,h})}{\sum_{h \in S} |y_h|}.$$

WQL evaluates the predicted distribution through its quantiles rather than only through a point forecast. At quantile level q, pinball loss penalizes underprediction with weight q and overprediction with weight $1 - q$. Across repeated forecast cases, its expected value is minimized when $Q_{q,h}$ is the conditional q-quantile of consumption. Averaging over $q = 0.1, \ldots, 0.9$ summarizes accuracy across the central predictive distribution.

The denominator $\sum_{h \in S} |y_h|$ scales the accumulated loss by the observed consumption in the evaluated hours. WQL is therefore unitless and more comparable across households with different consumption levels. Lower values indicate more accurate predictive quantiles, with zero attained only when every evaluated quantile equals the observation. A score is undefined only when its denominator is zero. We first calculate the ratio for each household–event pair and then average the defined ratios, so each pair contributes equally rather than in proportion to its total consumption. The Riemann model predicts $\widetilde{y}_h = \{\log(1 + y_h) - \mu_{\mathcal{C}}\}/\sigma_{\mathcal{C}}$. If $p_{\widetilde{Y}}$ is its predictive density on this normalized scale, its original-scale quantiles are $Q_{q,h} = \max\{0, \exp(\mu_{\mathcal{C}} + \sigma_{\mathcal{C}} Q^{\widetilde{Y}}_{q,h}) - 1\}$, and its original-scale NLL is $-\log p_{\widetilde{Y}}(\widetilde{y}_h) + \log \sigma_{\mathcal{C}} + \log(1 + y_h)$. We first average NLL over the hours in S and then over household–event pairs. Thus WQL and NLL use the same task-level aggregation.

We form 95% intervals with 2,000 crossed bootstrap draws that resample households and events independently. The primary metric is computed on active hours. High-only, Low-only, Normal-tariff, and whole-day scores report performance separately by tariff scope.

Seed zero's development set contains 1,827 defined active-hour WQL household–event scores. The direct PFN, unadapted Adaptive-PFN, and adapted Adaptive-PFN obtain 0.6972, 0.7160, and 0.4722, respectively. The paired adapted reduction over the unadapted Adaptive-PFN is 0.2437 (95% CI $[0.1592, 0.3509]$). These outcomes are used only to freeze the procedure; they are not pooled with the evaluation results.

Table 7 gives the primary active-hour comparison for each training seed. The procedure was fixed using the seed-zero development results before evaluation. Seeds 1 and 2 use the same evaluation households, events, and procedure without further tuning.

Table 7: **Active-hour London WQL across training seeds.** Lower WQL is better. Here $\Delta\mathrm{WQL} = \mathrm{WQL}_{\mathrm{calibrated}} - \mathrm{WQL}_{\mathrm{adapted}}$, so positive values favor the adapted Adaptive-PFN. Brackets give 95% intervals for this paired difference from 2,000 crossed household/event bootstrap draws for each seed, conditional on its fitted population tilt. Each row uses the same 5,865 defined active-hour household–event scores.

Seed	Direct	Calibrated	Unadapted	Adapted	ΔWQL: calibrated − adapted [95% CI]
0	0.8606	0.6760	0.8598	0.4957	0.1803 [0.0604, 0.3901]
1	0.8770	0.7768	0.7562	0.4798	0.2971 [0.0675, 1.2090]
2	0.8699	0.6818	0.9352	0.5965	0.0853 [−0.0580, 0.2293]

Deployment uncertainty. The intervals in Table 7 condition on the 248 population-fitting households used to estimate the prior tilt. We additionally form 200 joint replicates. Each replicate resamples the population-fitting households, recomputes the mean standardized semantic summary, refits $\widehat{\eta}$ against the fixed source bank, and evaluates the resulting frozen predictor after independently resampling evaluation households and tariff events. Evaluation outcomes never enter the tilt fit. Table 8 shows that the WQL reduction remains positive for every seed against both the unadapted Adaptive-PFN and the direct PFN. Defining $\Delta\mathrm{NLL} = \mathrm{NLL}_{\mathrm{comparator}} - \mathrm{NLL}_{\mathrm{adapted}}$, the corresponding NLL intervals are also entirely positive: their lower endpoints range from 0.210 to 0.253 against the unadapted Adaptive-PFN and from 0.145 to 0.197 against the direct PFN.

Table 8: **London WQL gains with deployment-population uncertainty.** Each difference is $\Delta\mathrm{WQL} = \mathrm{WQL}_{\mathrm{comparator}} - \mathrm{WQL}_{\mathrm{adapted}}$, so positive values favor the adapted Adaptive-PFN. Each entry gives the mean difference followed by its joint 95% interval. The resampling unit for tilt estimation is the population-fitting household; evaluation resamples evaluation households and events.

Seed	ΔWQL: unadapted − adapted	ΔWQL: direct − adapted
0	0.356 [0.208, 0.718]	0.357 [0.149, 1.021]
1	0.271 [0.152, 0.602]	0.387 [0.162, 1.346]
2	0.333 [0.190, 0.720]	0.268 [0.080, 0.811]

The calibrated direct PFN's active-hour NLL values are 625.27, 293.42, and 628.87 for seeds 0, 1, and 2, compared with -0.193, -0.231, and -0.217 for Adaptive-PFN. Their paired calibrated-minus-adapted differences have 95% intervals $[167.26, 1555.51]$, $[137.82, 512.00]$, and $[155.24, 1546.07]$.

The affine correction narrows the output distribution enough to improve WQL, but its contracted scale assigns very low density to active-hour observations far from its center. In this experiment, semantic-path reweighting improves both WQL and NLL. Table 9 reports tariff-scope scores for the seed-zero evaluation.

Table 9: **Seed-zero evaluation by tariff scope.** Lower is better. WQL is computed on the original consumption scale; "active" pools High and Low tariff hours, and "inactive" uses Normal-tariff hours in the same events. All neural networks are frozen and were trained only on simulated tasks. The primary comparison is active-hour WQL.

	WQL					Active NLL
Predictor	Active	High	Low	Inactive	Overall	
Direct PFN	0.8606	0.7837	0.6613	0.6727	1.0125	0.0546
Calibrated direct PFN	0.6760	0.5759	0.6065	0.6727	0.8871	625.2657
Adaptive-PFN, unadapted	0.8598	0.7430	0.7210	0.7105	1.0991	0.1043
Adaptive-PFN, adapted	**0.4957**	**0.4306**	**0.4811**	0.6850	**0.8450**	**-0.1927**

Table 10 checks whether the London gain persists when each seed uses fewer sampled semantic paths. For each prediction, T_1 draws 256 paths once, and the 64-path calculation uses the first 64 of those same paths.

Table 10: **Posterior-path sensitivity on the London evaluation tasks.** Here $\Delta\text{WQL} = \text{WQL}_{\text{unadapted}} - \text{WQL}_{\text{adapted}}$, so positive values favor adaptation. Brackets give 95% intervals for this paired difference using the same crossed household/event bootstrap as the primary analysis. Effective sample size (ESS) summarizes concentration of the self-normalized posterior-path weights; larger values indicate that more sampled paths materially contribute to the adapted mixture. The ESS mean and minimum are taken over evaluation household–event pairs; their maxima are the path counts shown in the second column.

Seed	Paths	Unadapted WQL	Adapted WQL	ΔWQL [95% CI]	Mean ESS	Min. ESS
0	64	0.9076	0.4876	0.4200 [0.2098, 1.1886]	5.94	1.22
0	256	0.8598	0.4957	0.3641 [0.2072, 0.8948]	22.15	10.57
1	64	0.7531	0.5142	0.2389 [0.1346, 0.5392]	3.46	1.01
1	256	0.7562	0.4798	0.2764 [0.1485, 0.7278]	11.84	3.99
2	64	1.0357	0.5853	0.4504 [0.1848, 1.5170]	8.35	1.81
2	256	0.9352	0.5965	0.3387 [0.1861, 0.8366]	32.18	16.29

F.7 Full-weight fine-tuning

A central advantage of Adaptive-PFN is that it adapts a pretrained model without updating network weights. We therefore compare it with a full-weight fine-tuning baseline that updates all 3,636,520 parameters of the direct PFN using the same 248 population-fitting households. Because its predictions can depend strongly on the stopping time, we report the trajectory rather than only one fine-tuned checkpoint.

The 336-hour population-fitting histories yield 989 causal forecast tasks. A task uses a 24-hour target beginning at 05:00 and the latest 48 earlier observed Normal-tariff hours as its data history; unobserved target hours are excluded from the Riemann negative-log-likelihood loss. Five-fold cross-validation holds out complete households. The initial selection compares learning rates 3×10^{-6}, 10^{-5}, and 3×10^{-5} at epochs 0, 1, 2, 4, 8, 16, and 32 using original-scale validation NLL. It selects 3×10^{-5} and epoch 32 for all three training seeds. The stopping-time ablation keeps that learning rate and extends the trajectories to epochs 64, 128, 256, and 512.

Each epoch selects one of the available tasks uniformly for every training household and shuffles the selected tasks. Optimization uses AdamW with batches of 32, weight decay 10^{-4}, and gradient clipping at one. For the final evaluation, we restart from the simulated-data checkpoint, update on all 248 population-fitting households, and evaluate each listed checkpoint on the unchanged evaluation households and events. Evaluation outcomes do not select a checkpoint.

Across seeds, mean training NLL decreases from -0.679 at epoch 64 to -1.492 at epoch 512, whereas mean validation NLL increases from -0.494 to 0.074. This reversal occurs in every seed: training NLL continues to improve after predictive performance on held-out households begins to deteriorate.

Table 11: **Stopping-time ablation for full-network fine-tuning.** Entries are active-hour WQL on the unchanged evaluation households and events; lower is better. Fine-tuning updates all direct PFN parameters using the 248 population-fitting households. Adaptive-PFN changes no network weights and therefore has no epoch index. Each entry uses the same 5,865 defined household–event scores.

Method or epoch	Seed 0	Seed 1	Seed 2	Mean
Fine-tuning, 32	0.3936	0.4066	0.4146	0.4049
Fine-tuning, 64	0.4292	0.4143	0.4254	0.4230
Fine-tuning, 128	0.4673	0.4621	0.4932	0.4742
Fine-tuning, 256	1.7965	2.9986	1.2509	2.0153
Fine-tuning, 512	3.7566	35.8154	3.9448	14.5056
Adaptive-PFN, adapted	0.4957	0.4798	0.5965	0.5240

Full-weight fine-tuning is included as a ceiling-style baseline rather than as the deployment procedure targeted by this paper: it updates all network parameters, whereas Adaptive-PFN introduces the estimated deployment prior without retraining the pretrained model. With early stopping, this more permissive procedure attains lower mean WQL than Adaptive-PFN at epochs 32, 64, and 128. Its advantage is not stable, however. Training NLL continues to decrease while validation NLL and evaluation WQL deteriorate, and by epoch 256 fine-tuning is worse than the original direct PFN for every seed. Thus, with only the population-fitting data available, full-weight fine-tuning can overfit and requires held-out households to choose when training should stop. Adaptive-PFN requires neither weight updates nor checkpoint selection.

G Rossmann sales experiment

In the Rossmann training simulator, the semantic path contains two real-valued coefficients, one for each promotion indicator used to generate log sales. Because this prior is only two-dimensional, we estimate its deployment distribution directly from store-level regressions rather than through the moment-matching procedure used in London. Adaptive-PFN then uses T_1 to apply this estimated prior when predicting for a new store.

G.1 Data and forecasting task

The Rossmann Store Sales dataset contains daily sales, opening status, promotion schedules and calendar information for 1,115 stores in Germany (Cukierski, 2015). We use the labeled records from 1 January 2013 through 31 July 2015. We do not use customer counts because they would not be known for a future forecast, and we do not use the competition test set.

One task corresponds to one store. Index its 56-day data history by $j = -56, \ldots, -1$ and its 42-day prediction period by $j = 0, \ldots, 41$. The data history ends on 19 June 2015, and the prediction period runs from 20 June through 31 July. Let $R_{ij}^{\text{raw}} \geq 0$ be the recorded sales of store i on day j, and let $O_{ij} \in \{0, 1\}$ indicate whether the store is open.

We first define the recorded calendar and promotion variables. Because all stores share the same forecast dates, the weekday and annual position depend on j but not on i. Let $d_j \in \{0, \ldots, 6\}$ equal zero on Monday, one on Tuesday, and so on through six on Sunday. Let $a_j \in \{0, \ldots, 365\}$ be the number of calendar days elapsed from January 1 of the same year to the date indexed by j; thus $a_j = 0$ on January 1 and $a_j = 1$ on January 2.

The remaining recorded variables can differ across stores. The indicator H_{ij}^{school} equals one when the dataset records that store i is affected by a public-school closure on day j. The recorded state-holiday category is

$$C_{ij}^{\text{state}} \in \{0, \text{a}, \text{b}, \text{c}\},$$

where 0 means no state holiday, a a public holiday, b an Easter holiday, and c Christmas. The network receives the binary indicator

$$H_{ij}^{\text{state}} = \mathbf{1}\{C_{ij}^{\text{state}} \neq 0\},$$

because the training simulator distinguishes only whether a state holiday occurs. The store-level regressions below instead retain the three recorded holiday categories so that their effects are not forced to be equal. The indicator P_{ij} is the dataset's daily `Promo` field: it equals one when store i is running a promotion on day j and zero otherwise. We call this an ordinary promotion to distinguish it from Promo2.

Promo2 requires a daily indicator constructed from store-level metadata. Let $A_{2i} \in \{0, 1\}$ be the dataset's `Promo2` field, which records whether store i participates in the Promo2 program. For a participating store, let τ_{2i} be the Monday of the ISO year and week recorded in `Promo2SinceYear` and `Promo2SinceWeek`, and let $\mathcal{M}_{2i}$ be the set of months listed in `PromoInterval`. If $\text{date}(j)$ and $\text{month}(j)$ denote the calendar date and month indexed by j, we define

$$P_{2,ij} = \begin{cases} 1, & A_{2i} = 1,\ \text{date}(j) \geq \tau_{2i},\ \text{and}\ \text{month}(j) \in \mathcal{M}_{2i}, \\ 0, & \text{otherwise.} \end{cases}$$

Thus $P_{2,ij} = 1$ precisely when Promo2 applies to store i on day j; it is not merely an indicator that the store participates in the program.

The model input x_{ij} has eleven coordinates. The first records the day's position relative to the forecast origin. The next two encode the weekday and the following two encode the position within the year. The sixth through tenth coordinates record the school holiday, state holiday, opening status, ordinary promotion, and daily Promo2 status. The final coordinate records the response scale estimated from the data history and is defined below. In this order,

$$x_{ij} = \Bigg(\frac{j+1}{56}, \sin\frac{2\pi d_j}{7}, \cos\frac{2\pi d_j}{7}, \sin\frac{2\pi a_j}{365.25}, \cos\frac{2\pi a_j}{365.25}, \\ H_{ij}^{\text{school}}, H_{ij}^{\text{state}}, O_{ij}, P_{ij}, P_{2,ij}, \phi_R(\sigma_i) \Bigg),$$

where the first coordinate is zero on the final day of the data history and $1/56$ on the first target day. The sine–cosine pairs represent weekday and annual position without a discontinuity between Sunday and Monday or between the end and beginning of a year.

The response is normalized using open days in the data history only. With

$$\mathcal{O}_i = \{j \in \{-56, \ldots, -1\} : O_{ij} = 1\}, \qquad r_{ij} = \log(1 + R_{ij}^{\text{raw}}),$$

define

$$\mu_i = \frac{1}{|\mathcal{O}_i|} \sum_{j \in \mathcal{O}_i} r_{ij}, \qquad \sigma_i = \min\left\{ 1.5, \max\left[0.05, \left\{ \frac{1}{|\mathcal{O}_i|} \sum_{j \in \mathcal{O}_i} (r_{ij} - \mu_i)^2 \right\}^{1/2} \right] \right\}.$$

The lower bound prevents division by a nearly zero standard deviation. The upper bound limits the influence of unusually variable data histories and matches the range used to encode the scale for the network. The normalized response is $\widetilde{y}_{ij} = (r_{ij} - \mu_i)/\sigma_i$, and the scale input is

$$\phi_R(\sigma) = 2\frac{\log\sigma - \log 0.05}{\log(1.5/0.05)} - 1.$$

Thus the observed data history is

$$D_i = \{(x_{ij}, \widetilde{y}_{ij}) : j \in \mathcal{O}_i\}.$$

In the fixed 56-row network input, rows for closed days are retained as padding but masked from the data-history encoding; their recorded zero sales are not treated as observations. For each target day, the query is x_{ij} and the response to be predicted is $\widetilde{y}_{ij}$. Predictions are returned to the sales scale through $\exp(\mu_i + \sigma_i\widetilde{y}) - 1$.

We retain stores with at least 14 open days in the data history. The prespecified primary evaluation set consists of open target days with an ordinary promotion, $\{j : O_{ij} = P_{ij} = 1\}$. Promo2 remains

an input, and its association remains part of the semantic path defined below, but Promo2 does not define the primary scoring set.

The eligible stores are assigned to three fixed, disjoint groups: 512 population-fitting stores, 256 development stores, and 256 evaluation stores. The evaluation stores are used only after the procedure has been fixed from the population-fitting and development groups.

G.2 SYNTHETIC TRAINING SIMULATOR

The networks are trained only on simulated stores. The simulated schedules provide variation in openings, holidays, and both promotion types, while the response model varies calendar effects, temporal dependence, and observation noise across tasks.

A simulated task contains $98 = 56 + 42$ consecutive days indexed by $k = 0, \dots, 97$. Draw an origin $o \sim \mathrm{Unif}\{0, \dots, 364\}$ and define the integer $c_k = o - 56 + k$. This integer fixes the phases of the simulated calendar coordinates; it does not denote a date in the observed dataset. Define the day of week $d_k = c_k \bmod 7$, the 30-day-block index $m_k = \lfloor c_k/30 \rfloor \bmod 12$, and the relative-time coordinate $t_k^{\mathrm{rel}} = (k - 55)/56$. These quantities determine the weekly, annual, and relative-time inputs used above.

The simulator draws the opening and promotion schedules as follows. It sets $H_k^{\mathrm{school}} = \mathbf{1}\{m_k \in \{6, 7, 11\}\}$ and independently draws $H_k^{\mathrm{state}} \sim \mathrm{Bernoulli}(0.02)$. A store closes on Sundays with probability 0.7: draw $C^{\mathrm{Sun}} \sim \mathrm{Bernoulli}(0.7)$. Independently draw $C_k^{\mathrm{holiday}} \sim \mathrm{Bernoulli}(0.8)$ for every day. Then

$$O_k = 1 - \mathbf{1}\Big\{(d_k = 6 \text{ and } C^{\mathrm{Sun}} = 1) \text{ or } (H_k^{\mathrm{state}} = 1 \text{ and } C_k^{\mathrm{holiday}} = 1)\Big\}.$$

The probability 0.02 gives an average of $98(0.02) = 1.96$ state-holiday days per simulated task. The Sunday and holiday closure probabilities allow some stores to remain open on those days, instead of imposing the same opening schedule on every task.

For ordinary promotions, draw a store-specific probability $p_P \sim \mathrm{Unif}[0.25, 0.55]$. Set $P_k = 0$ on closed days and Sundays; on each open Monday–Saturday, independently draw $P_k \sim \mathrm{Bernoulli}(p_P)$. If the resulting target period $k = 56, \dots, 97$ contains no ordinary-promotion day, set $P_k = 1$ on its first open Monday–Saturday in that period. This guarantees that the response to an ordinary promotion can be evaluated in every simulated target period. The interval $[0.25, 0.55]$ makes the ordinary-promotion frequency differ across stores: conditional on an open Monday–Saturday, the expected proportion of promoted days is p_P.

For Promo2, draw participation $A_2 \sim \mathrm{Bernoulli}(0.55)$, $h_2 \sim \mathrm{Unif}\{0, 1, 2\}$, which selects the value of $m_k \bmod 3$ for which Promo2 is active, and a start position $s_2 \sim \mathrm{Unif}\{-365, \dots, 55\}$. The Promo2 indicator is

$$P_{2k} = A_2\, \mathbf{1}\{k \geq s_2\} \mathbf{1}\{m_k \bmod 3 = h_2\} O_k.$$

The participation probability produces tasks both with and without Promo2. The start-position range allows Promo2 to begin before or during the data history, and $m_k \bmod 3 = h_2$ assigns each participating store one of three sequences of 30-day blocks.

The response components are drawn independently for each task. Draw

$$\gamma_0 \sim \mathrm{Unif}[7.5, 9.5], \qquad \widetilde{w}_0, \dots, \widetilde{w}_6 \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 0.15^2),$$

$$w_d = \widetilde{w}_d - \frac{1}{7}\sum_{h=0}^{6} \widetilde{w}_h, \qquad \gamma_{\sin}, \gamma_{\cos}, \gamma_{\mathrm{trend}} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 0.12^2),$$

$$\rho \sim \mathrm{Unif}[0, 0.8], \qquad \sigma_\epsilon \sim \mathrm{Unif}[0.08, 0.25].$$

The centered values w_d describe day-of-week differences without changing the store level γ_0. The standard deviations 0.15 and 0.12 set the magnitudes of the weekday, annual, and trend terms on the log-sales scale. For temporal dependence, draw the stationary AR(1) process

$$v_0 \sim \mathcal{N}(0, 1), \qquad v_k = \rho v_{k-1} + \sqrt{1 - \rho^2}\, \xi_k, \qquad \xi_k \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1).$$

Finally draw $\epsilon_k \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1)$. The interval $[0, 0.8]$ makes the lag-one correlation of the AR(1) process range from zero to 0.8, while $\sigma_\epsilon \in [0.08, 0.25]$ varies the standard deviation of the independent

observation noise across tasks. The coefficient 0.12 in Equation 78 fixes the marginal standard deviation of the AR(1) term.

The semantic path contains only the two promotion associations,

$$M = (\beta_P, \beta_2), \qquad \beta_P, \beta_2 \overset{\text{ind}}{\sim} \mathrm{TN}(0, 0.5^2; [-1, 1]).$$

This prior permits positive and negative promotion associations, assigns them standard deviation 0.5 before truncation, and bounds each association between -1 and 1 on the log-sales scale. Conditional on these draws, the unnormalized log-sales response is

$$\begin{aligned} r_k = \gamma_0 + w_{d_k} + \gamma_{\sin} \sin \frac{2\pi c_k}{365.25} + \gamma_{\cos} \cos \frac{2\pi c_k}{365.25} + \gamma_{\text{trend}} t_k^{\text{rel}} + 0.12 v_k + \sigma_\epsilon \epsilon_k \\ + \beta_P P_k + \beta_2 P_{2k}. \end{aligned} \tag{78}$$

The store level, calendar coefficients, trend, AR(1) process, observation noise, and simulated schedules vary across tasks but are not coordinates of M. The two promotion associations are therefore available for prior reweighting. T_2 receives them together with the observed data history and target-day inputs, but it does not receive the store level, calendar coefficients, trend, AR(1) process, or noise draws separately.

As for the observed stores, simulated responses are centered and scaled using open days in the data history only, with standard deviation floored at 0.05; closed responses are masked. The scale input uses ϕ_R, clipped to $[-1, 1]$ if the scale computed from the simulated data history lies outside $[0.05, 1.5]$. A simulated task must contain at least 14 open days in its data history and at least one open target day.

Both semantic-path coordinates are active in T_1. Let F_0 be the cumulative distribution function of $\mathrm{TN}(0, 0.5^2; [-1, 1])$. Internally, T_1 generates

$$U_P = F_0(\beta_P), \qquad U_2 = F_0(\beta_2),$$

which are independent uniform variables on $[0, 1]$ under the training prior. The inverse transformation recovers β_P and β_2 from the generated coordinates. The numerical representation supplied to T_2 is $c(m) = (\beta_P, \beta_2) \in [-1, 1]^2$.

We train one Adaptive-PFN and one direct PFN with approximately equal parameter counts using seed zero. The semantic posterior network T_1 and conditional predictive network T_2 contain 1,827,265 and 1,804,516 parameters. Their total of 3,631,781 differs by 0.15% from the direct PFN's 3,637,096 parameters. We train T_1 and T_2 for 4,096 updates each and the direct PFN for 8,192 updates, with batches of 128 independently simulated tasks. The direct PFN therefore sees the same total number of training tasks as T_1 and T_2 combined. Optimization uses AdamW, peak learning rate 3×10^{-4}, weight decay 10^{-4}, gradient clipping at one, linear warm-up over the first $1/16$ of the updates, and cosine decay thereafter.

All predictive distributions use the same 100-bucket Riemann predictive distribution with half-normal tails. Its 101 bucket borders are estimated once from the open target responses of 100,000 independently simulated tasks and shared by Adaptive-PFN and the direct PFN. On 512 independent tasks from the training prior, scoring all open target days and using 128 paths for Adaptive-PFN, the unadapted Adaptive-PFN and direct PFN obtain NLL 1.2468 and 1.2429.

G.3 Deployment prior estimation

The deployment prior in this experiment is the across-store distribution of the two promotion associations. We estimate this distribution from the population-fitting stores, without using either PFN. For store i, let $b_i = (\beta_{P,i}, \beta_{2,i})^\top$ denote the coefficients of the ordinary- promotion and Promo2 indicators in the calendar-adjusted sales regression defined below. The remaining regression terms account for trend and recurring calendar variation in that store's sales. For every open day t before 20 June 2015, define the regression row

$$\begin{aligned} u_{it}^{\text{reg}} = \Bigg(1, \frac{t - t_*}{365.25}, \sin \frac{2\pi a_t}{365.25}, \cos \frac{2\pi a_t}{365.25}, \mathbf{1}\{d_t = 1\}, \ldots, \mathbf{1}\{d_t = 6\}, \\ H_{it}^{\text{school}}, \mathbf{1}\{C_{it}^{\text{state}} = \text{a}\}, \mathbf{1}\{C_{it}^{\text{state}} = \text{b}\}, \mathbf{1}\{C_{it}^{\text{state}} = \text{c}\}, P_{it}, P_{2,it}\Bigg)^\top, \end{aligned}$$

where t_* is 20 June 2015 and $t - t_*$ is the signed number of calendar days between the two dates. Here d_t and a_t are the weekday and number of days since January 1 for date t, defined as above. Monday is the omitted weekday, and a day without a state holiday is the omitted state-holiday category. The three separate state-holiday indicators retain distinctions that are collapsed into one network input. Let U_i collect these rows, let n_i be their number, and let r_i collect the corresponding values of $\log(1 + R_{it}^{\mathrm{raw}})$.

A population-fitting store is eligible when it has at least 365 open pre-forecast observations, at least 20 open observations with and without an ordinary promotion, and at least 20 open observations with and without Promo2. Thus each promotion coefficient is estimated from at least 20 rows with its indicator equal to one and at least 20 rows with its indicator equal to zero. Of the 512 population-fitting stores, $N = 263$ satisfy them.

For an eligible store, let

$$\Lambda = \mathrm{diag}(0, 10^{-8}, \ldots, 10^{-8})$$

and compute

$$\widehat{\theta}_i = (U_i^\top U_i + \Lambda)^+ U_i^\top r_i,$$

where the superscript $+$ denotes the Moore–Penrose inverse. Define

$$\widehat{\sigma}_i^2 = \max\left\{ \frac{\|r_i - U_i\widehat{\theta}_i\|_2^2}{n_i - \mathrm{rank}(U_i^\top U_i)}, 0.05^2 \right\}, \qquad \widehat{V}_i = \widehat{\sigma}_i^2 (U_i^\top U_i + \Lambda)^+.$$

The final two entries of $\widehat{\theta}_i$ form $\widehat{b}_i = (\widehat{\beta}_{P,i}, \widehat{\beta}_{2,i})^\top$, and the corresponding 2×2 block $\widehat{V}_i^{(P,2)}$ estimates their sampling covariance.

The fitted vectors $\widehat{b}_i$ vary across stores for two reasons. The promotion associations b_i themselves differ across stores, and each $\widehat{b}_i$ is estimated from a finite and noisy sales history. The difference $\widehat{b}_i - b_i$ is the resulting coefficient-estimation error. Under the working least-squares model,

$$\mathbb{E}[\widehat{b}_i - b_i \mid U_i] = 0, \qquad \mathrm{Cov}(\widehat{b}_i - b_i \mid U_i) \approx \widehat{V}_i^{(P,2)}.$$

Consequently, if the promotion-association vectors have across-store mean μ and covariance Σ, then

$$\mathbb{E}[\widehat{b}_i] = \mu, \qquad \mathrm{Cov}(\widehat{b}_i) \approx \Sigma + \mathbb{E}[\widehat{V}_i^{(P,2)}].$$

The sample covariance of the fitted coefficients therefore combines genuine between-store variation with coefficient-estimation error. Following the random-coefficient method-of-moments correction of Swamy (1970, Section 2, Equation (2.12), p. 315), we estimate the former by

$$\widehat{\mu} = \frac{1}{N} \sum_{i=1}^{N} \widehat{b}_i,$$

$$\widehat{\Sigma}_{\mathrm{raw}} = \frac{1}{N-1} \sum_{i=1}^{N} (\widehat{b}_i - \widehat{\mu})(\widehat{b}_i - \widehat{\mu})^\top - \frac{1}{N} \sum_{i=1}^{N} \widehat{V}_i^{(P,2)}.$$

The second term removes the estimated contribution of coefficient-estimation error to the observed across-store covariance. This is a method-of-moments estimator because it matches the empirical covariance of $\widehat{b}_i$ to the decomposition above and solves for Σ. After symmetrizing $\widehat{\Sigma}_{\mathrm{raw}}$, we eigendecompose it and replace any eigenvalue below 0.02^2 by 0.02^2. The floor prevents a negative measurement-error-corrected eigenvalue from producing an invalid covariance matrix and places a lower bound of 0.02 on the estimated between-store standard deviation in any direction. Denote the resulting matrix by $\widehat{\Sigma}$. The estimated across-store location and measurement-error-corrected covariance are

$$\widehat{\mu} = \begin{pmatrix} 0.3526 \\ -0.0145 \end{pmatrix}, \qquad \widehat{\Sigma} = \begin{pmatrix} 0.007983 & -0.000303 \\ -0.000303 & 0.001126 \end{pmatrix}. \tag{79}$$

We use $\widehat{\mu}$ and $\widehat{\Sigma}$ as the location and pre-truncation covariance parameters of a Gaussian restricted to the training support $[-1, 1]^2$, denoted by $\widehat{p}_*$. The fitted location for β_P is 0.3526 rather than the training-prior location zero; the fitted location for β_2 is -0.0145.

G.4 APPLYING THE DEPLOYMENT PRIOR

We now use $\widehat{p}_*$ to predict for a development or evaluation store i. The store's 56-day data history D_i first enters T_1. Using $S = 512$, draw

$$M_i^{(1)}, \ldots, M_i^{(S)} \overset{\text{i.i.d.}}{\sim} T_1(dm \mid D_i).$$

These draws approximate the semantic-path posterior under the training prior p_0. To replace that prior by $\widehat{p}_*$, assign each path the density ratio

$$\alpha_i^{(s)} = \frac{\widehat{p}_*(M_i^{(s)})}{p_0(M_i^{(s)})}, \qquad \widetilde{\alpha}_i^{(s)} = \frac{\alpha_i^{(s)}}{\sum_{r=1}^{S} \alpha_i^{(r)}}. \tag{80}$$

For target day j, T_2 produces one predictive distribution conditional on each sampled path. Their weighted mixture is

$$\widehat{q}_*(dy \mid x_{ij}, D_i) = \sum_{s=1}^{S} \widetilde{\alpha}_i^{(s)} T_2(dy \mid x_{ij}, D_i, M_i^{(s)}). \tag{81}$$

The prior shift in this application is the change from p_0 to $\widehat{p}_*$. No value of η is estimated: the store-level regressions estimate $\widehat{p}_*$ directly, and the prior ratio is evaluated for every path drawn by T_1. This is the independently estimated-prior case described after Equation 4, rather than the moment-matching procedure in Equation 6.

We also check the reweighting calculation on 512 simulated tasks. The two promotion associations are drawn independently from Gaussians with locations $\widehat{\mu}$ and marginal variances given by the diagonal of $\widehat{\Sigma}$; every other draw follows the training simulator. Each prediction uses 128 paths. On open ordinary-promotion target days, reweighting reduces predictive-mean MSE on the normalized log-response scale from 0.7919 to 0.7127 and NLL on the same scale from 1.2854 to 1.2380. This check verifies the prior-ratio calculation under a known shift; unlike the deployment prior, it does not include the off-diagonal covariance in $\widehat{\Sigma}$.

G.5 BASELINES, METRICS, AND RESULTS

We compare four predictors, all with fixed network parameters. The *direct PFN* is the single network trained on the simulator. The *unadapted Adaptive-PFN* averages T_2 using the paths returned by T_1 without reweighting, whereas the *adapted Adaptive-PFN* uses the density-ratio weights in Equation 80. The fourth predictor, the *calibrated direct PFN*, is an output-only baseline: it uses the population-fitting stores to change only its predicted distribution, by estimating one additive log-sales translation for each promotion indicator.

We estimate this translation using the population-fitting stores and an earlier forecasting window that does not overlap the development or evaluation targets. For each population-fitting store, a 56-day data history ending on 8 May 2015 is used to predict 9 May through 19 June. Let $\mathcal{I}_{\text{fit}}$ denote the population-fitting stores, let $\mathcal{T}_{\text{cal}}$ denote these 42 target dates, and define

$$\mathcal{C} = \{(i,t) : i \in \mathcal{I}_{\text{fit}},\ t \in \mathcal{T}_{\text{cal}},\ O_{it} = 1,\ P_{it} = 1 \text{ or } P_{2,it} = 1\}.$$

Thus $\mathcal{C}$ contains the open target days on which at least one promotion is active. For $(i,t) \in \mathcal{C}$, let $\widehat{r}_{it}^{\text{direct}}$ be the direct PFN's predicted median of $\log(1 + R_{it}^{\text{raw}})$ from the corresponding earlier data history, and define $e_{it} = \log(1 + R_{it}^{\text{raw}}) - \widehat{r}_{it}^{\text{direct}}$ and $v_{it} = (P_{it}, P_{2,it})^\top$. We estimate one additive correction for each promotion type by

$$\widehat{\delta} = (\widehat{\delta}_P, \widehat{\delta}_2)^\top = \arg\min_{\delta \in \mathbb{R}^2} \left\{ \sum_{(i,t)\in\mathcal{C}} (e_{it} - v_{it}^\top \delta)^2 + 10^{-8} \|\delta\|_2^2 \right\}.$$

The quadratic penalty regularizes this two-coefficient least-squares fit. If, for any $q \in (0,1)$, $Q_{q,ij}^{\log,\text{direct}}$ is the direct PFN's predicted q-quantile on the log-sales scale for a development or evaluation target day, we set $Q_{q,ij}^{\log,\text{cal}} = Q_{q,ij}^{\log,\text{direct}} + P_{ij}\widehat{\delta}_P + P_{2,ij}\widehat{\delta}_2$. The same translation is applied

to the entire log-sales density. It changes the predicted location on promotion days but leaves the shape and dispersion of the direct PFN's distribution unchanged. Neither this calibration nor the prior replacement in Adaptive-PFN updates any network parameter.

For an evaluated store i, let $\mathcal{A}_i = \{j \in \{0, \dots, 41\} : O_{ij} = 1,\ P_{ij} = 1\}$ be its open ordinary-promotion target days. WQL and NLL are computed on this set, while RMSPE uses its positive-sales subset defined below. Let $\mathcal{Q} = \{0.1, 0.2, \dots, 0.9\}$, $Q_{q,ij}$ be the predicted q-quantile on the original sales scale, and $\rho_q(u) = u\{q - \mathbf{1}(u < 0)\}$ be the pinball loss. We instantiate the WQL defined in Appendix F.6 for store i as

$$\mathrm{WQL}_i = \frac{1}{9} \sum_{q \in \mathcal{Q}} \frac{2 \sum_{j \in \mathcal{A}_i} \rho_q(R_{ij}^{\mathrm{raw}} - Q_{q,ij})}{\sum_{j \in \mathcal{A}_i} R_{ij}^{\mathrm{raw}}}.$$

This score evaluates the nine predicted quantiles and divides their loss by the store's observed sales on the evaluated days. The reported WQL is the arithmetic mean of WQL_i across stores, so every store has equal weight.

We additionally report RMSPE because it is the conventional point-forecast metric for the Rossmann dataset. Unlike WQL, it evaluates only the predictive median and therefore shows whether an improvement in the predicted distribution is also present in its central forecast. Define

$$\mathcal{A}_i^+ = \{j \in \mathcal{A}_i : R_{ij}^{\mathrm{raw}} > 0\}, \qquad \mathrm{RMSPE}_i = \left[\frac{1}{|\mathcal{A}_i^+|} \sum_{j \in \mathcal{A}_i^+} \left(\frac{Q_{0.5,ij} - R_{ij}^{\mathrm{raw}}}{R_{ij}^{\mathrm{raw}}} \right)^2 \right]^{1/2}.$$

Zero-sales days are excluded because their percentage error is undefined. Squaring gives greater weight to large relative errors, and the square root returns the result to the relative-error scale. We report the arithmetic mean of RMSPE_i across stores. Thus, for example, an RMSPE of 0.13642 is a store-averaged RMSPE of 13.642%, not a mean relative error.

Finally, if p_{ij}^{norm} is a predictor's density for the normalized log response, its original-sales density and store-level negative log-likelihood are

$$p_{ij}^{\mathrm{sales}}(R) = \frac{p_{ij}^{\mathrm{norm}}(\{\log(1+R) - \mu_i\}/\sigma_i)}{\sigma_i(1+R)},$$

$$\mathrm{NLL}_i = -\frac{1}{|\mathcal{A}_i|} \sum_{j \in \mathcal{A}_i} \log p_{ij}^{\mathrm{sales}}(R_{ij}^{\mathrm{raw}}).$$

The denominator in p_{ij}^{sales} is the Jacobian of the transformation from the normalized log response back to sales. Reported NLL is the arithmetic mean of NLL_i across stores.

We form the reported WQL intervals from 1,000 paired crossed-bootstrap replicates. Let $\mathcal{W}$ be the set of ISO year–week pairs represented in the 42 target dates. Each replicate samples the 256 stores in the evaluated group with replacement and, independently, $|\mathcal{W}|$ weeks with replacement. The weight of a store–week pair is the product of its two resampling counts. The same weights are used for both predictors in a comparison, so each interval describes a paired metric difference.

We proceed to final evaluation only if adapted WQL improves over both the unadapted Adaptive-PFN and the direct PFN with paired 95% intervals excluding zero, and adapted NLL is no higher than either reference. On the development stores, adapted WQL is 0.09706, compared with 0.10333 for the unadapted Adaptive-PFN and 0.10231 for the direct PFN. The 95% interval for adapted-minus-direct WQL is $[-0.00749, -0.00269]$; the interval against the unadapted Adaptive-PFN also excludes zero, and adapted NLL is lower than both values. The criterion is therefore met, and we apply the same procedure to the evaluation stores without further changes.

Table 12 reports that evaluation. The adapted Adaptive-PFN improves WQL, RMSPE, and NLL relative to each of the other three predictors. Relative to the direct PFN, WQL decreases by 7.48%; relative to the unadapted Adaptive-PFN, it decreases by 7.51%. The 95% intervals for the adapted-minus-reference WQL differences are $[-0.00982, -0.00511]$ and $[-0.01094, -0.00357]$, respectively.

Table 12: **Rossmann evaluation results.** Metrics use 3,840 open ordinary-promotion days from 256 evaluation stores; lower is better. RMSPE is reported as a proportion, so 0.10 corresponds to 10%. The calibrated direct PFN estimates two promotion-specific translations from the population-fitting stores and applies them to its log-sales distribution.

Method	WQL	RMSPE	NLL
Direct PFN	0.09855	0.14726	8.6110
Adaptive-PFN, unadapted	0.09858	0.14603	8.6071
Calibrated direct PFN	0.10601	0.15932	8.6668
Adaptive-PFN, adapted	**0.09118**	**0.13642**	**8.5320**

The adapted prediction uses 512 semantic paths sampled from $T_1(dm \mid D_i)$ for each store. Their posterior-path effective sample size, defined in Equation 26, has mean 94.1 and minimum 7.0 across the evaluation stores. Recomputing the prediction from the first 128 sampled paths changes WQL from 0.09118 to 0.09139, an absolute difference of 0.00021. This is small relative to the 0.00737 WQL difference between the adapted Adaptive-PFN and the direct PFN.