What does a Bayes-filtered transformer believe? A predictive Monte Carlo approach

Afiq Abdillah Effiezal Aswadi* **Haotong Ma*** **Susan Wei**
Department of Econometrics and Business Statistics
Monash University

Abstract

A *Bayes-filtered transformer* (BFT) is a transformer trained on sequences that are generated in two steps: first a latent task is drawn from a prior, then observations are drawn conditional on that task. Trained under autoregressive log loss, the BFT's next-token prediction, in the idealized limit, *is* the Bayesian posterior predictive distribution (PPD) induced by that prior and that conditional law. In practice the trained BFT is only an approximation of this ideal PPD, raising an interpretive question: what prior and posterior over the latent task has the trained BFT actually internalized? Existing work answers this question by comparing the trained BFT's predictions against the predictions of various baseline posteriors, each standing in for a different candidate algorithm or computation the BFT might be implementing. This prediction-space comparison is fragile: different posteriors can share the same posterior-mean predictions. We use *predictive Monte Carlo* (PMC) as an interpretability tool for BFTs: using only next-token generation, we obtain with PMC an approximation to the implicit prior and posterior over the latent task, answering the interpretive question directly in latent space. We apply PMC to three stylized task families spanning 0-Markov and 1-Markov exchangeability. We reproduce previously reported phenomena in latent space and show that comparisons of PMC priors and posteriors can reveal patterns in model behavior that the corresponding prediction-space comparisons miss. Code is available at https://github.com/afiq-aswadi/bft-pmc.

1 Introduction

In many studies of in-context learning, transformers are trained on sequences generated in two steps. A latent task is drawn from a prior, and observations are drawn conditional on the task. Autoregressive log-loss training on such sequences is minimized by the Bayesian posterior predictive distribution (PPD) [31]. We introduce the term *Bayes-filtered transformer* (BFT) for a transformer trained in this way. In the idealized limit of sufficient capacity, infinite data, and perfect optimization, a BFT *is* this PPD. In practice, the trained BFT realizes only some approximation of the ideal object.

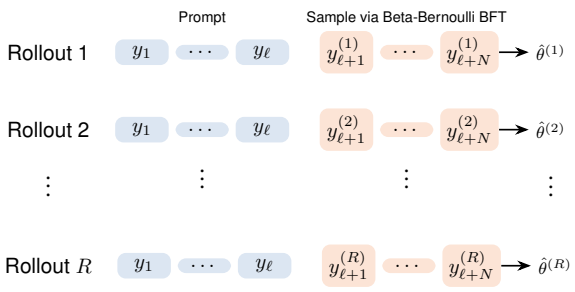
An interpretive question follows: given a trained BFT, what prior and posterior belief over the latent task has it actually internalized? *Predictive Monte Carlo* (PMC) [14], a technique from Bayesian predictive inference (BPI) [15], can be used to answer this question using only next-token generation. With PMC, we obtain approximate samples from the BFT's implicit prior over the latent task by rolling out from an empty prompt, and from its implicit posterior by rolling out from a given prompt. In each case, the draw of the latent task is computed from the completed rollout, for instance as an empirical frequency or a regression fit.

As a first illustration, consider a BFT pretrained on binary sequences generated from a Bernoulli likelihood with a uniform $\mathrm{Beta}(1,1)$ prior on the success probability θ, a *Beta-Bernoulli BFT*.

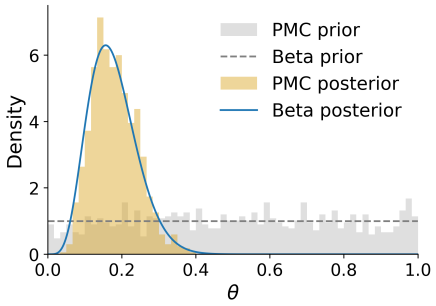
*Equal contribution.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Figure 1 shows PMC applied to such a model. Given a prompt $y_{1:\ell}$, the PMC procedure (panel a) draws R rollouts from the Beta-Bernoulli BFT and, for each rollout r, computes the sample mean $\hat{\theta}^{(r)}$ of the completed sequence. The empirical distribution of $\{\hat{\theta}^{(r)}\}_{r=1}^{R}$ is the BFT's PMC posterior over θ. Running the same procedure with an empty prompt gives the PMC prior. Panel (b) compares the PMC prior samples with the $\mathrm{Beta}(1,1)$ prior used during pretraining, and the PMC posterior samples with the conjugate Beta posterior. Despite being only an approximation to the Beta-Bernoulli PPD, the PMC prior and posterior over θ are close to the $\mathrm{Beta}(1,1)$ prior and the conjugate posterior, respectively. Agreement of both prior and posterior in latent space suggests that the model encodes the pretraining prior and updates it via Bayes' rule. A prediction-space comparison can establish neither, since distinct priors and posteriors can produce identical predictions.



(a) PMC procedure requires only forward passes.



(b) PMC prior and posterior samples vs the analytic Beta prior and conjugate posterior.

Figure 1: **PMC schematic.** Given a (possibly empty) prompt $y_{1:\ell}$, the Beta-Bernoulli BFT autoregressively generates R rollouts $y_{\ell+1:\ell+N}^{(r)}$ of length N (panel a, adapted from Ng et al. [29]). For each rollout r, $\hat{\theta}^{(r)} = \frac{1}{\ell+N}\left(\sum_{i=1}^{\ell} y_i + \sum_{i=\ell+1}^{\ell+N} y_i^{(r)}\right)$ is the sample mean of the full rollout (including the initial prompt). Panel (b) shows the histogram of $\{\hat{\theta}^{(r)}\}_{r=1}^{R}$ for an empty prompt and for a non-empty prompt. With the *empty* prompt, the $\hat{\theta}^{(r)}$ form the PMC *prior*, an approximation to the BFT's implicit prior over θ. The PMC prior tracks the $\mathrm{Beta}(1,1)$ prior used during pretraining. With the *non-empty* prompt (here $\ell = 32$ with 5 ones and 27 zeros), they form the PMC *posterior*, which tracks the conjugate posterior $\mathrm{Beta}(6,28)$. Training and PMC settings, and the comparison for nine prompts (Figure 5), are in Appendix B.

BFTs are not limited to illustrations such as the Beta-Bernoulli example. Prior-data fitted networks (PFNs) [26] are BFTs trained on exchangeable sequences. TabPFN [21, 22] and TabICL [34] are PFNs pretrained on synthetic tabular datasets. They predict on real tabular data by **in-context learning** (ICL) alone, with no parameter updates at inference time. TabPFN is currently state of the art on small tabular classification and regression problems [22], and TabICL extends in-context classification to tables with hundreds of thousands of rows [34]. Predictive resampling, the procedure underlying PMC, has already been applied to TabPFN [29].

In a parallel line of work, authors use BFTs as a controlled testbed for studying ICL in transformers. Transformers trained on synthetic pretraining distributions learn simple function classes in context [18, 32], including linear regression [40, 43] and Markov chains [11]. When the posterior under the pretraining distribution is available in closed form, the transformer's predictions can be compared with the Bayesian posterior predictive distribution. In another group of studies, the authors fix two baseline priors. The **generalizing** prior Π_∞ is a continuous prior on the latent task. The **memorizing** prior Π_M is the uniform distribution on M pretraining tasks drawn independently from Π_∞. The number M is the *task diversity*. The transformer is then characterized by which of the two induced posteriors more closely describes it. Raventós et al. [35] make this comparison for linear regression at the end of training, and Carroll et al. [7] make it over the course of training. Wurgaft et al. [41] make it for additional exchangeable data-generating mechanisms, including balls-and-urns, and Park et al. [33] for mixtures of Markov chains. We take these three settings, *balls-and-urns*, *linear regression*, and *Markov chains* (Table 1), as our task families. Two empirical phenomena have emerged:

- **Task-diversity threshold** [35]: as the task diversity M grows, the transformer shifts from the memorizing scheme to the generalizing scheme, despite the memorizing scheme being optimal under the pretraining distribution.
- **Transient generalization** [7]: at intermediate task diversity, the transformer first approximates the generalizing scheme during training before later specializing to the memorizing scheme.

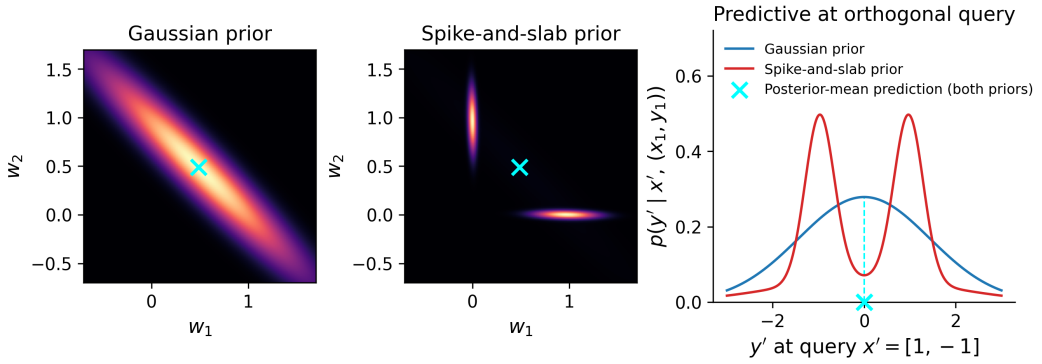


Figure 2: **Prediction-space measurement is fragile.** 2-D Bayesian linear regression: posterior over the coefficient $w \in \mathbb{R}^2$ given a single observation $(x_1, y_1) = ([1, 1], 1)$, computed under two different priors. The Gaussian and spike-and-slab priors yield posteriors that share the mean $(0.5, 0.5)$ (marked $\times$) but differ in shape (left, middle). At the test query $x' = [1, -1]$, the two predictive distributions over y' are unimodal vs bimodal (right), yet the posterior-mean predictions coincide. A comparison of posterior means therefore cannot distinguish the two posteriors.

Carroll et al. [7], Park et al. [33], Raventós et al. [35], Wurgaft et al. [41] measure the BFT in prediction space. They compare its predictions with those of the memorizing and generalizing schemes, either at the end of training as task diversity M varies, or throughout training at a fixed M. The discrepancy measure varies with the setting, from MSE between point predictions [7, 35] to KL-type divergences between next-token predictives [11, 33, 41]. The two schemes differ in their posteriors over the latent task. Distinct posteriors can produce identical predictions, so a prediction-space comparison cannot always tell the two schemes apart. Figure 2 gives a concrete example for MSE: two priors with the same posterior mean give identical point predictions even when their posterior shapes are markedly different. Divergences between next-token predictives are more discriminating, but they depend on the posterior only through the predictive. In the categorical families, the predictive at each position is determined by the posterior mean of the latent parameter. Distinct posteriors with the same posterior-mean trajectory are therefore indistinguishable in prediction space.

Contributions. **(i)** We unify previously disparate synthetic ICL studies (balls-and-urns [41], linear regression [7, 35], and Markov chains [11, 33]) as k-*Markov exchangeable* BFTs. **(ii)** We move ICL evaluation from prediction space into latent space, repurposing PMC as a black-box interpretability tool for k-Markov exchangeable BFTs that uses only next-token generation. **(iii)** To our knowledge, this is the first application of PMC to BFTs. **(iv)** Across the three task families, we reproduce the phenomena of task-diversity threshold and transient generalization in latent space and identify patterns in model behavior that the corresponding prediction-space comparisons miss.

Outline. In Section 2 we define the exchangeability properties (0-Markov and 1-Markov) of our task families. In Section 3 we define meta-learning on Bayes-filtered data. Its idealized limit is the Bayesian PPD that a trained BFT approximates. In Section 4 we present PMC and sufficient conditions for its validity. In Section 5 we define the memorizing and generalizing baselines and the prediction-space and latent-space metrics, and we apply PMC to balls-and-urns, linear regression, and Markov chains. In Section 6 we discuss related work on ICL phenomenology and on Bayesian predictive inference. A table of notation is in Appendix A.

2 Markov exchangeability

We focus on three settings from the ICL literature: balls-and-urns [41], linear regression [7, 35], and finite-state Markov chains [11, 33]. We unify them under a single framework, k-Markov

exchangeability. Under each of the three pretraining distributions, the observation sequence is k-Markov exchangeable for some k. For linear regression, the observation sequence is the sequence of pairs (X_i, Y_i). For balls-and-urns and linear regression, $k = 0$, which is exchangeability (Section 2.1). For Markov chains, $k = 1$ (Section 2.2). The definition for general k is in Appendix C. The latent task, kernel and prior of each family are in Table 1.

Throughout, $\mathcal{Y}$ denotes the observation space. It is a finite set for balls-and-urns and Markov chains, and $\mathcal{Y} = \mathbb{R}$ for the response in linear regression. Write $\mathcal{Y}^m$ for the set of length-m sequences over $\mathcal{Y}$, with $\mathcal{Y}^0 := \{\varnothing\}$, and $\mathcal{Y}^* := \bigcup_{m\geq 0} \mathcal{Y}^m$ for the set of finite sequences. We use $\mathcal{Y}^\infty$ to denote the set of infinite sequences. We work with a sequence of random variables $(Y_n)_{n\geq 1}$ taking values in $\mathcal{Y}$, with joint law $\mathbb{P}$, and with the associated sequence of one-step predictive distributions $(P_n)_{n\geq 0}$, called the **predictive rule**,

$$P_0(A) := \mathbb{P}(Y_1 \in A), \qquad P_n(A) := \mathbb{P}(Y_{n+1} \in A \mid Y_{1:n}), \qquad A \subseteq \mathcal{Y},\ n \geq 1. \tag{1}$$

The controlled ICL setups we consider all draw sequences from a pretraining distribution of the same form: a finite-dimensional random **latent task** $\tilde{\theta} \in \Theta$ is drawn from a prior Π, and tokens are then generated sequentially via a probability kernel $P_{\tilde{\theta}}$ from $\mathcal{Y}^*$ to $\mathcal{Y}$, meaning that after history $h \in \mathcal{Y}^*$ the next token is drawn from $P_{\tilde{\theta}}(\cdot \mid h)$. The history argument lets the next token depend on earlier tokens, as in the Markov chains family. The resulting joint law is

$$\mathbb{P}(Y_{1:n} = y_{1:n}) = \mathbb{E}_{\tilde{\theta}\sim\Pi}\left[\prod_{i=1}^{n} P_{\tilde{\theta}}(y_i \mid y_{1:i-1})\right]. \tag{2}$$

The corresponding one-step predictive, written in terms of the random prefix $Y_{1:n}$ rather than a fixed value $y_{1:n}$, is

$$P_n(A) = \mathbb{E}_{\tilde{\theta}\sim\Pi(\cdot\mid Y_{1:n})}\big[P_{\tilde{\theta}}(A \mid Y_{1:n})\big], \qquad A \subseteq \mathcal{Y}, \tag{3}$$

where $\Pi(\cdot \mid Y_{1:n})$ is the posterior on Θ. $\tilde{\theta}$ and $P_{\tilde{\theta}}$ for each family are in Table 1.

2.1 Exchangeability (0-Markov exchangeability)

A sequence $(Y_n)_{n\geq 1}$ on $\mathcal{Y}$ with joint law $\mathbb{P}$ is *infinitely exchangeable* (*exchangeable*, for short) if $(Y_1, \ldots, Y_n) \overset{d}{=} (Y_{\sigma(1)}, \ldots, Y_{\sigma(n)})$ for every n and every permutation σ of $\{1, \ldots, n\}$. By **de Finetti's representation theorem** [8, 15], exchangeability is equivalent to the existence of a random probability measure $\tilde{P}$ on $\mathcal{Y}$ such that conditional on $\tilde{P}$ the tokens $Y_1, Y_2, \ldots$ are i.i.d. with distribution $\tilde{P}$, which is called the **directing random measure** of the exchangeable sequence.

In balls-and-urns, the tokens are exchangeable. The latent task is a probability vector $\tilde{p} \in \Delta^{|\mathcal{Y}|-1}$, the probability simplex on $\mathcal{Y}$, with prior $\Pi(\tilde{p}) = \mathrm{Dirichlet}(1, \ldots, 1)$, and the directing random measure is $\tilde{P} = \mathrm{Categorical}(\tilde{p})$: conditional on $\tilde{p}$, the tokens are i.i.d. $\mathrm{Categorical}(\tilde{p})$. In linear regression, the exchangeable observations are the pairs $Z_i = (X_i, Y_i)$. The latent task is the regression vector $\tilde{w} \in \mathbb{R}^d$, with prior $\Pi(\tilde{w}) = \mathcal{N}(0, I_d)$. Conditional on $\tilde{w}$, the pairs are i.i.d. with $X_i \sim \mathcal{N}(0, I_d)$ and $Y_i \mid X_i \sim P_{\tilde{w}}(\cdot \mid X_i)$. Here $P_{\tilde{w}}(\cdot \mid x) = \mathcal{N}(\tilde{w}^\top x, \sigma^2)$ with $\sigma^2 = 0.25$ (Appendix G).

2.2 1-Markov exchangeability

The general k-Markov case ($k \geq 2$) is in Appendix C. For an exchangeable sequence, the law of a path depends only on how many times each token value appears in it. The Markov analog lets the law depend on the path's initial state and its transition counts. Formally, a sequence $(Y_n)_{n\geq 1}$ on $\mathcal{Y}$ with joint law $\mathbb{P}$ is *1-Markov exchangeable* [15] if for every n and any two paths $y_{1:n}, y'_{1:n} \in \mathcal{Y}^n$,

$$y_1 = y'_1 \text{ and } N_{a,b}(y_{1:n}) = N_{a,b}(y'_{1:n})\ \forall a, b \in \mathcal{Y} \implies \mathbb{P}(Y_{1:n} = y_{1:n}) = \mathbb{P}(Y_{1:n} = y'_{1:n}),$$

where $N_{a,b}(y_{1:n}) := \sum_{i=1}^{n-1} \mathbf{1}\{y_i = a,\ y_{i+1} = b\}$ are the transition counts.

Let $\mathcal{Q} := \{Q = (Q_{a,b})_{a,b\in\mathcal{Y}} :\ Q_{a,b} \in [0,1],\ \sum_{b\in\mathcal{Y}} Q_{a,b} = 1\ \forall a \in \mathcal{Y}\}$ denote the set of transition matrices on $\mathcal{Y}$. Under recurrence (the initial state is visited infinitely often, $\mathbb{P}$-almost surely), the **Diaconis–Freedman representation theorem** [10] says that Markov exchangeability is equivalent to the existence of a **random transition matrix** $\tilde{Q} \in \mathcal{Q}$, with prior distribution Π on $\mathcal{Q}$, such that conditional on $\tilde{Q}$, (Y_n) is a Markov chain governed by the same transition matrix $\tilde{Q}$ at every step.

For a transition matrix $Q \in \mathcal{Q}$, the entry $Q_{a,b}$ is the probability of moving from state a to state b, and $Q_{a,\cdot}(A) := \sum_{b \in A} Q_{a,b}$ is the probability that the next state lies in A given current state a. The one-step predictive is the PPD $P_n(A) = \mathbb{E}_{\tilde{Q} \sim \Pi(\cdot \mid Y_{1:n})}[\tilde{Q}_{Y_n,\cdot}(A)]$ for $A \subseteq \mathcal{Y}$, where the posterior on $\tilde{Q}$ is $\Pi(\mathrm{d}Q \mid Y_{1:n}) \propto \prod_{i=1}^{n-1} Q_{Y_i,Y_{i+1}}\, \Pi(\mathrm{d}Q)$, with Y_1 taken as given so that $P(Y_1 \mid \tilde{Q})$ does not appear in the likelihood. A 1-Markov exchangeable law of the form (2) has latent task $\tilde{Q}$ and kernel $P_{\tilde{Q}}(y \mid y_{1:n}) = \tilde{Q}_{y_n,y}$. This kernel depends on the history only through its last token. Our Markov chains family [11, 33] takes the prior Π under which the rows of $\tilde{Q}$ are independent and each row $(\tilde{Q}_{a,b})_{b \in \mathcal{Y}}$, $a \in \mathcal{Y}$, has the $\mathrm{Dirichlet}(1, \ldots, 1)$ distribution.

Table 1: **Synthetic task families used in this paper.** The first two rows are exchangeable (0-Markov) families. The third row instantiates 1-Markov exchangeability, following Park et al. [33]. $\mathcal{Q}$ is the set of transition matrices on $\mathcal{Y}$, that is, $|\mathcal{Y}| \times |\mathcal{Y}|$ matrices with nonnegative entries and unit row sums.

Synthetic task family	Latent task $\tilde{\theta}$	Kernel	Prior
Balls-and-urns (App. H)	$\tilde{\theta} = \tilde{p} \in \Delta^{\lvert\mathcal{Y}\rvert-1}$	$Y_i \mid \tilde{p} \sim \mathrm{Categorical}(\tilde{p})$	$\Pi(\tilde{p}) = \mathrm{Dirichlet}(1, \ldots, 1)$
Linear regression (App. G)	$\tilde{\theta} = \tilde{w} \in \mathbb{R}^d$	$X_i \sim \mathcal{N}(0, I_d)$, $Y_i \mid X_i, \tilde{w} \sim \mathcal{N}(\tilde{w}^\top X_i, \sigma^2)$	$\Pi(\tilde{w}) = \mathcal{N}(0, I_d)$
Markov chains (App. I)	$\tilde{\theta} = \tilde{Q} \in \mathcal{Q}$	$Y_i \mid Y_{1:i-1}, \tilde{Q} \sim \tilde{Q}_{Y_{i-1},\cdot}$	$\Pi(\tilde{Q}) =$ row-wise $\mathrm{Dirichlet}(1, \ldots, 1)$

3 Meta-learning on Bayes-filtered data

We present the meta-learning framework that leads to the definition of the *Bayes-filtered transformer* (Definition 1). We develop it in the unsupervised setting (token sequences $Y_{1:n}$ on $\mathcal{Y}$, covering balls-and-urns and Markov chains); linear regression is recovered by replacing Y_i with the pair (X_i, Y_i), with covariate $X_i \in \mathcal{X}$.

Pretraining distribution. The meta-learning setup has two ingredients, both from Section 2: a prior Π on the latent task $\tilde{\theta} \in \Theta$ and, for each $\theta \in \Theta$, a probability kernel P_θ from $\mathcal{Y}^*$ to $\mathcal{Y}$. A pretraining sequence is drawn by sampling $\tilde{\theta} \sim \Pi$ and then generating $Y_{1:n}$ token by token from $P_{\tilde{\theta}}$. The joint law of a pretraining sequence is then $\mathbb{P}$ of (2).

From Bayesian prediction to optimization. Let $T_\phi \colon \mathcal{Y}^* \to \Delta(\mathcal{Y})$ be a sequence model, e.g. a transformer, where $\Delta(\mathcal{Y})$ denotes the set of probability distributions on $\mathcal{Y}$. By the Ionescu–Tulcea theorem [23], T_ϕ induces a unique joint law $\mathbb{P}_\phi$ on $\mathcal{Y}^\infty$ whose one-step predictives are T_ϕ at every history, so $\mathbb{P}_\phi(Y_{1:n} = y_{1:n}) = \prod_{i=1}^{n} T_\phi(y_i \mid y_{1:i-1})$. We fit T_ϕ to $\mathbb{P}$ by minimizing the KL between the two joint laws:

$$\arg\min_\phi \ \mathrm{KL}\big(\mathbb{P}(Y_{1:n}) \,\|\, \mathbb{P}_\phi(Y_{1:n})\big) \ = \ \arg\min_\phi \ \mathbb{E}_{\mathbb{P}}\left[-\sum_{i=1}^{n} \log T_\phi(Y_i \mid Y_{1:i-1})\right], \tag{4}$$

where the equality uses the factorization of $\mathbb{P}_\phi$ and drops the entropy of $\mathbb{P}$ (independent of ϕ). The result is the standard autoregressive log loss, which converts the computation of the Bayesian PPD of the pretraining model $(\Pi, \{P_\theta\}_{\theta \in \Theta})$, typically intractable, into an optimization problem. (For linear regression, T_ϕ predicts Y_i from the prefix $(X_{1:i}, Y_{1:i-1})$. Its output is a density on $\mathbb{R}$, a mixture over 256 equal-width bins covering $[-10, 10]$ that is uniform on each inner bin and half-normal on the two edge bins, which extend to $\pm\infty$ (Appendix G).)

Per-token targets are Bayes-filtered. At every position i, the objective (4) fits T_ϕ to predict Y_i given the realized prefix $Y_{1:i-1}$. Under the pretraining distribution, the conditional law of Y_i given $Y_{1:i-1}$ is the Bayesian PPD,

$$\mathbb{P}(Y_i \in A \mid Y_{1:i-1}) = \mathbb{E}_{\tilde{\theta} \sim \Pi(\cdot \mid Y_{1:i-1})}\big[P_{\tilde{\theta}}(A \mid Y_{1:i-1})\big], \qquad A \subseteq \mathcal{Y}, \tag{5}$$

with $\Pi(\cdot \mid Y_{1:i-1})$ the posterior induced by Π and the observed prefix. Each pretraining target is thus *filtered* through Bayes' rule against the latent task. Ortega et al. [31] call such data *Bayes-filtered*.

Meta-learning [31]. *Meta-learning* is empirical risk minimization of (4) on a pretraining set $\mathcal{D}$ of sequences drawn from $\mathbb{P}$:

$$\arg\min_{\phi} \frac{1}{|\mathcal{D}|} \sum_{y_{1:n} \in \mathcal{D}} \Big[-\sum_{i=1}^{n} \log T_\phi(y_i \mid y_{1:i-1}) \Big]. \tag{6}$$

In our experiments, each training batch is a fresh draw from $\mathbb{P}$, so no sequence is seen twice.

Definition 1 (Bayes-filtered transformer)**.** *A* Bayes-filtered transformer *(BFT) is a transformer T_ϕ fit by the meta-learning objective* (6) *on sequences drawn from the pretraining joint law $\mathbb{P}$ of* (2)*, for some choice of kernel family $\{P_\theta\}_{\theta \in \Theta}$ and prior Π.*

In the idealized limit of sufficient capacity, infinite pretraining data, and perfect optimization, the trained BFT equals the Bayesian PPD (5) at every prefix of length less than n. In practice these conditions fail, so a trained BFT is only an approximation to the PPD. PMC (Section 4) is designed to answer the interpretive question of what prior and posterior over θ the trained BFT has actually internalized.

The transformers of Carroll et al. [7], Edelman et al. [11], Park et al. [33], Raventós et al. [35], Wurgaft et al. [41] are BFTs in the sense of Definition 1, each for one of the families of Section 2. Our output head for linear regression differs from that of Raventós et al. [35] and Wurgaft et al. [41] (Appendix F).

4 Predictive Monte Carlo (PMC)

Predictive Monte Carlo (PMC) [14] is a procedure that can be used to sample from the prior and the posterior implied by a predictive rule. The implicit prior and the implicit posterior of a BFT are well defined when the observation sequence, under the joint law that the BFT induces, is exchangeable, asymptotically exchangeable, or k-Markov exchangeable and recurrent for some $k \geq 1$. We define the implicit prior and the implicit posterior below. We then describe how PMC is applied to T_ϕ to sample from the implicit prior and the implicit posterior.

Let $\mathbb{P}_\phi$ be the joint law on $\mathcal{Y}^\infty$ whose one-step predictives are T_ϕ (Section 3). If $(Y_n)_{n\geq 1}$ is infinitely exchangeable under $\mathbb{P}_\phi$, we write $\tilde{P}_\phi$ for its directing random measure (Section 2.1). If $(Y_n)_{n\geq 1}$ is 1-Markov exchangeable and recurrent under $\mathbb{P}_\phi$, we write $\tilde{Q}_\phi$ for its random transition matrix (Section 2.2). The law of $\tilde{P}_\phi$, or of $\tilde{Q}_\phi$, is the **implicit prior** of the BFT. Its conditional law given a prompt $y_{1:\ell}$ is the **implicit posterior** of the BFT.

Let $\mathcal{P}$ be the set of probability measures on $\mathcal{Y}$. Let θ be a functional of interest, defined on $\mathcal{P}$ or on $\mathcal{Q}$ and taking values in $\mathbb{R}^D$ for some $D \geq 1$. The law of $\theta(\tilde{P}_\phi)$, or of $\theta(\tilde{Q}_\phi)$, is the pushforward of the implicit prior under θ. We call it the implicit prior of θ. Its conditional law given $y_{1:\ell}$ is the implicit posterior of θ. In each family, we take θ to be the map to the latent task. This map is well defined because the kernel family of each task family is identifiable. That is, distinct latent tasks give distinct kernels. In general, θ need not be the latent task of a model [13].

- **Balls-and-urns.** $\theta(\tilde{P}_\phi) = (\tilde{P}_\phi(\{a\}))_{a\in\mathcal{Y}}$, the vector $\tilde{p}$ with $\tilde{P}_\phi = \mathrm{Categorical}(\tilde{p})$.
- **Linear regression.** $\theta(\tilde{P}_\phi) = \big(\mathbb{E}_{\tilde{P}_\phi}[XX^\top]\big)^{-1}\mathbb{E}_{\tilde{P}_\phi}[XY]$, where $(X, Y) \sim \tilde{P}_\phi$.
- **Markov chains.** $\theta(\tilde{Q}_\phi) = \tilde{Q}_\phi$.

By exchangeability, the empirical distributions of $Y_{1:n}$ converge weakly to $\tilde{P}_\phi$, $\mathbb{P}_\phi$-almost surely [15, Theorem 2.2]. By 1-Markov exchangeability and recurrence, the normalized transition counts converge to $\tilde{Q}_\phi$, $\mathbb{P}_\phi$-almost surely [10]. Then $\theta(\tilde{P}_\phi)$ and $\theta(\tilde{Q}_\phi)$ are functions of the infinite path $y_{1:\infty}$, and we write each as $\theta(y_{1:\infty})$.

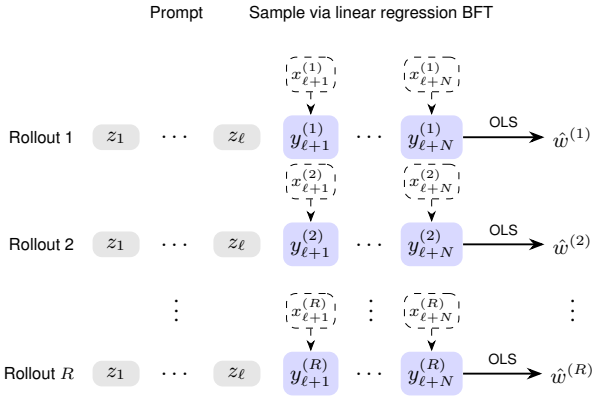
- **Balls-and-urns.** $\theta(y_{1:\infty})_a = \lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}\{y_i = a\}$ for each $a \in \mathcal{Y}$.
- **Linear regression.** $\theta(y_{1:\infty}) = \lim_{n\to\infty}\big(\sum_{i=1}^{n} x_i x_i^\top\big)^{-1}\sum_{i=1}^{n} x_i y_i$.
- **Markov chains.** $\theta(y_{1:\infty})_{a,b} = \lim_{n\to\infty} N_{a,b}(y_{1:n}) \big/ \sum_{b'\in\mathcal{Y}} N_{a,b'}(y_{1:n})$ for all $a, b \in \mathcal{Y}$.

With PMC, we construct a **single sample** of the latent task given $y_{1:\ell}$ in two steps:

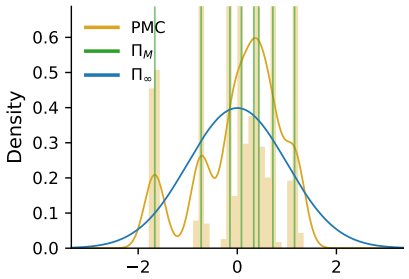
(i) **Rollout**: extend $y_{1:\ell}$ by iteratively drawing $Y_{\ell+1} \sim T_\phi(\cdot \mid y_{1:\ell})$, $Y_{\ell+2} \sim T_\phi(\cdot \mid y_{1:\ell+1})$, and so on. By construction, the completed path is distributed according to $\mathbb{P}_\phi$ conditioned on the path beginning with $y_{1:\ell}$.

(ii) **Evaluate**: compute $\theta(y_{1:\infty})$ on the completed path. The draw has the conditional law of $\theta(y_{1:\infty})$ given $y_{1:\ell}$ under $\mathbb{P}_\phi$ (the implicit prior when $y_{1:\ell}$ is empty, the implicit posterior otherwise).

Step (i) cannot be carried out to infinity, so each rollout is truncated at finite length N, and the infinite-path limits above are replaced by finite-rollout estimators: empirical token frequencies for balls-and-urns (Appendix H), ordinary least squares on the pair sequence for linear regression (Appendix G), and Laplace-smoothed empirical transition counts for Markov chains, where the smoothing keeps every row of the estimated matrix well-defined when a state is rarely visited within the truncated rollout (Appendix I). To produce R approximate samples of the latent task given $y_{1:\ell}$, we repeat the rollout-and-evaluate construction R times independently, at $R \cdot N$ forward passes through T_ϕ in total (Figure 3a for linear regression). We write $\widehat{\Pi}(\cdot \mid y_{1:\ell})$ for the empirical distribution of the R samples. We call it the **PMC posterior**. With an empty prompt, $\widehat{\Pi}(\cdot \mid \varnothing)$ is the **PMC prior**.



(a) PMC procedure for linear regression.



(b) PMC prior of the $M = 8$ linear regression BFT versus the training prior Π_M and the population prior Π_∞.

Figure 3: **PMC for in-context linear regression.** (a) The latent task is the regression coefficient $w \in \mathbb{R}^d$. Unlike the Beta-Bernoulli setting of Figure 1, each rollout step draws a covariate $x_t \sim \mathcal{N}(0, I_d)$ and then a response $y_t \sim T_\phi(\cdot \mid x_t, z_{1:t-1})$, where $z_t = (x_t, y_t)$. The rollout estimator $\hat{w}^{(r)}$ is OLS on the completed sequence $z^{(r)}_{1:\ell+N}$. (b) PMC applied with an empty prompt to the linear regression BFT trained at task diversity $M = 8$. The histogram shows the PMC samples of the first coordinate of w. The green vertical lines mark the first coordinates of the $M = 8$ pretraining tasks, the atoms of Π_M. The height of these lines has no meaning. Overlaid are the corresponding marginal densities of the training prior Π_M and of the population prior $\Pi_\infty = \mathcal{N}(0, I_d)$: the PMC samples concentrate near the former. The same comparison for four coordinates of w, with density and CDF panels, is in Figure 17. The full grid over task diversities is at `https://bft-pmc.github.io/` (Appendix J).

We end this section by noting that we do not in fact expect a trained BFT to be exchangeable or k-Markov exchangeable, because these symmetry properties are very strict. A trained BFT may not be exchangeable, but it may be asymptotically exchangeable. Following Fortini and Petrone [15], $(Y_n)_{n\geq 1}$ is **asymptotically exchangeable** under $\mathbb{P}_\phi$ with limit directing random measure $\tilde{P}_\phi$ if, as $n \to \infty$, $(Y_{n+1}, Y_{n+2}, \dots)$ converges in distribution to an exchangeable sequence with directing random measure $\tilde{P}_\phi$. That is, for large n the sequence behaves like an exchangeable sequence with directing random measure $\tilde{P}_\phi$. The implicit prior and implicit posterior are defined from $\tilde{P}_\phi$ as in the exchangeable case. Sufficient conditions for asymptotic exchangeability are conditions on the predictives $T_\phi(\cdot \mid Y_{1:n})$. These conditions, and recent attempts to verify them empirically for TabPFN, are in Appendix D. Under each of them, the predictives and the empirical distributions of $Y_{1:n}$ converge to $\tilde{P}_\phi$, $\mathbb{P}_\phi$-almost surely. So $\theta(y_{1:\infty})$ is defined as above. We do not verify these conditions for the BFTs in this paper. As an empirical proxy, we check whether estimates of the latent task, computed on sequences generated from T_ϕ, stabilize as the sequence length grows (Figures 7–9). To our knowledge, no asymptotic counterpart of k-Markov exchangeability has been formulated,

so for the Markov chain family PMC relies on exact 1-Markov exchangeability and recurrence. Formulating such a counterpart, with sufficient conditions on the predictives, is left open.

5 Experiments

We train BFTs in the three task families, apply PMC to each trained BFT, and compare the PMC prior and posterior with two baselines, in both prediction space and latent space. The main text shows figures for linear regression. The figures for balls-and-urns and Markov chains are in Appendices H and I. For each family we report three results: agreement of the PMC prior and posterior with the baselines, the task-diversity threshold, and transient generalization. The PMC settings, including R and N, are in Appendices G–I. Their effect, and that of the latent task dimension, on the PMC samples is studied in Appendix E.

Training transformers across task families. Architectural and training details are collected in Appendix F, with family-specific instantiations in Appendices G, H, and I. Following Raventós et al. [35], we vary the task diversity M. For each task family and each value of M, we form the empirical prior $\Pi_M := \mathrm{Unif}(\{\theta^{(1)}, \dots, \theta^{(M)}\})$ by drawing a fresh set $\theta^{(1)}, \dots, \theta^{(M)} \overset{\text{iid}}{\sim} \Pi_\infty$ once. The transformer is then trained on sequences drawn by sampling $\tilde{\theta} \sim \Pi_M$ and then generating $Y_{1:n}$ from $P_{\tilde{\theta}}$. We write $\mathbb{P}_M$ for the joint law (2) with prior Π_M.

Memorizing and generalizing baselines. The trained BFT is compared with two baselines. Let $c = y_{1:\ell}$ be an observed prompt. The *memorizing* baseline uses the empirical prior Π_M. The *generalizing* baseline uses the population prior Π_∞. The corresponding posteriors are

$$\Pi^{\mathrm{mem}}(\theta \mid c) \;\propto\; \Pi_M(\theta) \prod_{i=1}^{\ell} P_\theta(y_i \mid y_{1:i-1}), \qquad \Pi^{\mathrm{gen}}(\theta \mid c) \;\propto\; \Pi_\infty(\theta) \prod_{i=1}^{\ell} P_\theta(y_i \mid y_{1:i-1}). \quad (7)$$

For linear regression, c also contains the covariates $x_{1:\ell}$, and $P_\theta(y_i \mid y_{1:i-1})$ is replaced by $P_w(y_i \mid x_i)$. The covariate density does not depend on w, so it does not appear in (7). Each posterior induces a PPD for the next token. We derive the two posteriors and their PPDs for each family in Appendices G–I.

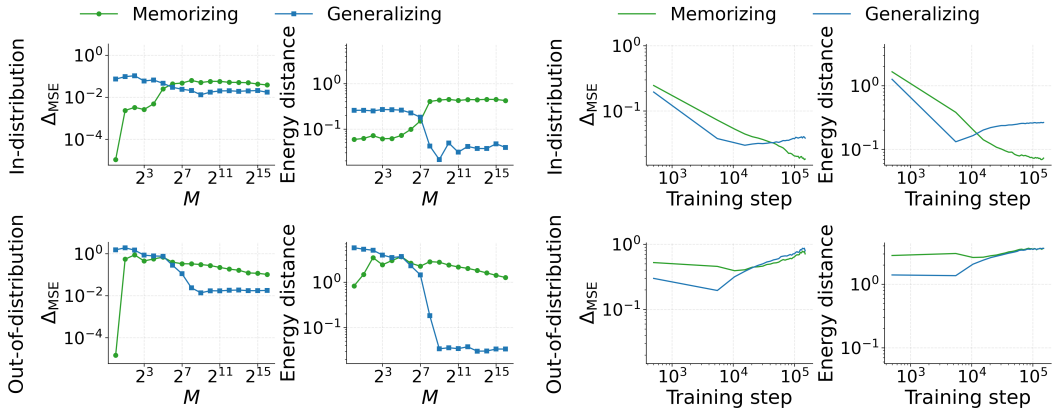
Evaluation metrics. In **prediction** space, we compare the BFT with each baseline using mean squared error for linear regression and symmetrized KL divergence for balls-and-urns and Markov chains. In **latent** space, we compare the PMC posterior $\widehat{\Pi}(\cdot \mid c)$ with $\Pi^{\mathrm{mem}}(\cdot \mid c)$ and with $\Pi^{\mathrm{gen}}(\cdot \mid c)$ using squared energy distance and sliced Wasserstein-1. Smaller values indicate closer agreement with the baseline. Every prompt has length $\ell = 32$ (for linear regression, a prompt is the pair sequence $z_{1:\ell}$). An *in-distribution* prompt is generated from P_θ with $\theta \sim \Pi_M$. An *out-of-distribution* prompt is generated from P_θ with $\theta \sim \Pi_\infty$. With probability one, this θ is none of the M pretraining tasks. These two kinds of prompt follow the evaluation on tasks seen during pretraining and on new tasks in Raventós et al. [35]. We average posterior latent-space distances separately over 128 prompts from each source. For the prior, we compare the PMC prior $\widehat{\Pi}(\cdot \mid \varnothing)$ with Π_M and with Π_∞. Metric definitions, estimators, and the rationale for these choices are given in Appendix F.

Linear regression (0-Markov exchangeable). At task diversity $M = 8$, the PMC prior marginals match the training prior Π_M (Figure 3b). At low task diversity M, the priors Π_M and Π_∞ are well separated, and the PMC prior matches Π_M. At high task diversity, Π_M approaches Π_∞, and the distances from the PMC prior to the two baselines become harder to distinguish.

For in-distribution prompts (top rows of Figure 4), at $M = 8$ the PMC posterior samples closely match the closed-form memorizing posterior (Figure 17). As M increases, the PMC posterior moves away from the memorizing posterior and toward the generalizing posterior (Figure 4(a)), reproducing the task-diversity threshold of Raventós et al. [35] in the posterior. At intermediate $M = 32$, the BFT first approximates the generalizing baseline and later moves toward the memorizing baseline (Figure 4(b)), reproducing the transient generalization of Carroll et al. [7] in both prediction space (left column) and in the posterior (right column).

For out-of-distribution prompts (bottom rows of Figure 4), the PMC posterior is far from both baselines at low and intermediate task diversity, and close to the generalizing posterior at high task diversity (Figure 4(a)). The posterior samples at $M = 8$ show this for individual coordinates of w (Figure 17). At $M = 32$, the BFT's predictions and the PMC posterior are close to the generalizing

baseline early in training, and far from both baselines at the end of training (Figure 4(b)). A BFT trained at task diversity M is pretrained on sequences generated by M tasks. A prompt generated by a task drawn from Π_∞ has small probability under $\mathbb{P}_M$, so such prompts are rare in pretraining.



(a) Task-diversity sweep at end of training.

(b) Training dynamics at $M = 32$.

Figure 4: **Linear regression: task-diversity threshold and transient generalization.** Each panel plots the distance to the memorizing baseline (green) and to the generalizing baseline (blue): against task diversity M at the end of training in (a), and against training step at $M = 32$ in (b). Rows: in-distribution prompts (top) and out-of-distribution prompts (bottom). Columns: Δ_{MSE} between the BFT's predictions and the baseline predictions, and energy distance between the PMC posterior and the baseline posterior. Both axes are log-scaled in every panel. The corresponding latent-space comparisons using sliced Wasserstein-1 are provided in Appendix G: Figure 18 for (a) and Figure 19 for (b).

Balls-and-urns (0-Markov exchangeable). At task diversity $M = 8$, the PMC prior marginals match the training prior Π_M (Figure 21, Appendix H). Conditioning on an in-distribution prompt of length 32, the PMC posterior samples match the closed-form memorizing posterior. On an out-of-distribution prompt of the same length, we see a similar result. We reproduce the task-diversity threshold in the posterior (Figure 22, Appendix H). The prediction-space view shows this transition only at larger task diversities, as we discuss in Appendix H. At task diversity $M = 32$, we also reproduce transient generalization (Figure 23, Appendix H).

Markov chains (1-Markov exchangeable). At task diversity $M = 4$, the PMC prior marginals match the training prior Π_M (Figure 24, Appendix I). Conditioning on an in-distribution prompt of length 32, the PMC posterior samples match the memorizing posterior. Conditioning on an out-of-distribution prompt of the same length, we see a similar result. This match is less apparent than in the other task families, as we discuss in Appendix I. We reproduce the task-diversity threshold in the posterior (Figure 25, Appendix I). At task diversity $M = 8$, we also reproduce transient generalization for in-distribution prompts in prediction space and under sliced Wasserstein, but not under energy distance (Figure 26, Appendix I).

6 Related work

The Bayesian view of ICL. Explanations of ICL in LLMs [6] include Bayesian accounts [42], induction heads [30], and implicit gradient descent [1, 38]. We follow the Bayesian line that posits the transformer's next-token predictions arise from a posterior over latent task structure updated by the context. Xie et al. [42] introduced this view as an explanation for ICL in pretrained LLMs, and Falck et al. [12] tested it empirically on LLMs through the martingale property. Our work moves to the narrower setting of synthetic studies, where transformers are trained on hierarchical pretraining distributions. Within this synthetic setting, Garg et al. [18] and Panwar et al. [32] show that transformers can learn simple function classes in context. Zhang et al. [43] show that a linear self-attention layer trained by gradient flow learns linear regression in context. Wu et al. [40] bound the number of pretraining tasks this requires. Follow-up work [7, 11, 33, 35, 41] asks which Bayesian

scheme the trained transformer adopts: a memorizing one (using the empirical pretraining prior) or a generalizing one (using the population prior).

BFTs. Ortega et al. [31] established the equivalence between meta-learning on hierarchical data and Bayesian posterior prediction. We adopt this view in Section 3, where the idealized limit of a BFT is the Bayesian PPD. Genewein et al. [19], Grau-Moya et al. [20], Mikulik et al. [25] show empirically that trained sequence models approach the Bayesian PPD on controlled synthetic tasks. Prior-data fitted networks [PFNs; 26] are BFTs trained on exchangeable data. Hollmann et al. [21, 22] scale this construction into TabPFN, a tabular foundation model.

BPI. Fortini and Petrone [14] introduced PMC, which uses a sequence's predictive distribution to draw samples from the prior or posterior over its directing random measure. Fortini and Petrone [15] survey the surrounding predictive framework. Fong et al. [13] use the same procedure, under the name *predictive resampling*, in the *martingale posterior* framework, a generalized Bayes setting in which the target functional need not index the probability model. Battiston and Cappello [2] weaken the c.i.d. condition, a sufficient condition for BPI, to almost c.i.d. Fortini et al. [16] weaken it to a quasi-martingale condition. Ng et al. [29] use TabPFN as the predictive rule of the martingale posterior (TabMGP). Fortini et al. [16] also use TabPFN as the predictive rule. They develop an asymptotic uncertainty decomposition for TabPFN under the quasi-martingale condition. A subsequent version of this work [17] relaxes the quasi-martingale to a weaker *signed* condition.

Related interpretability work. Shai et al. [37] study transformers trained on sequences from a hidden Markov model (HMM) data generating process and show that the Bayesian posterior over the HMM's hidden state is represented in the residual stream, by training a linear probe against the ground-truth posterior. Their setup shows the posterior is encoded within the model but does not provide a way to recover it without access to the ground truth. Whether PMC samples relate to the belief-state geometry [37] is a direction for future work.

7 Conclusion

Summary. We repurpose PMC as a black-box interpretability tool for BFTs. Using only next-token generation, we sample from the prior and posterior over the latent task that the BFT has internalized. Identifiability motivates moving the evaluation of ICL from prediction space to latent space: distinct posteriors can produce identical predictions, so prediction-space diagnostics cannot tell them apart. On balls-and-urns, linear regression, and Markov chains, we reproduce the task-diversity threshold and transient generalization directly in latent space.

Limitations. The sufficient conditions for asymptotic exchangeability are hard to verify for BFTs, both theoretically and empirically. Theoretically, we cannot currently show anything substantive about a real trained transformer, including the sufficient conditions for asymptotic exchangeability from the BPI literature. Empirically, the brute-force quasi-martingale tests of Fortini et al. [16] on TabPFN demand Monte Carlo budgets so large that verification has so far been inconclusive. Closing the gap needs weaker sufficient conditions for asymptotic exchangeability and empirical diagnostics with tractable Monte Carlo cost.

PMC also requires more computation than prediction-space evaluation. A prediction-space comparison requires a single forward pass per evaluation prompt, whereas PMC requires $R \cdot N$ forward passes to generate R rollouts of length N. Because the rollouts are independent, they can run in parallel. With sufficient hardware, generation takes N sequential rounds, each advancing all R rollouts by one step, while the total number of model evaluations remains $R \cdot N$. The direct applicability of PMC to LLMs is unclear, though not implausible: there is a relationship between BFTs and Solomonoff induction [20], and plausible arguments link LLMs to Solomonoff induction as well [9].

Outlook. Even absent direct transfer to frontier LLMs, BFTs are appealing as a model system for phenomena that in LLMs are entangled with many confounding factors. Emergent misalignment [3] is a concrete candidate: narrow finetuning on misbehaved examples induces broadly misaligned behavior, a mechanism difficult to isolate at scale and already suspected to operate through shifts in a latent persona representation [39]. A BFT whose prior predictive encodes a mixture of behavioral personas would offer a controlled setting in which the posterior over personas can be sampled with PMC before, during, and after exposure to adversarial examples. The same template extends to other LLM-scale phenomena whose underlying belief dynamics are suspected but not directly observable.

Acknowledgements

This work was supported by The Alignment Project (grant AP-S2-100069), funded by the UK AI Security Institute, Department for Science, Innovation and Technology. This work was supported by Monash eResearch capabilities, including M3.

References

[1] Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=0g0X4H8yN4I.

[2] Marco Battiston and Lorenzo Cappello. Bayesian predictive inference beyond martingales. *Journal of Machine Learning Research*, 2026. ISSN 1532-4435.

[3] Jan Betley, Daniel Chee Hian Tan, Niels Warncke, Anna Sztyber-Betley, Xuchan Bao, Martín Soto, Nathan Labenz, and Owain Evans. Emergent misalignment: Narrow finetuning can produce broadly misaligned LLMs. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu, editors, *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 4043–4068. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/betley25a.html.

[4] Sid Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, USVSN Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. GPT-NeoX-20B: An open-source autoregressive language model. In *Proceedings of the ACL Workshop on Challenges & Perspectives in Creating Large Language Models*, 2022. URL https://arxiv.org/abs/2204.06745.

[5] Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. Sliced and Radon Wasserstein Barycenters of Measures. *Journal of Mathematical Imaging and Vision*, 51 (1):22–45, 2015. ISSN 1573-7683. doi: 10.1007/s10851-014-0506-3. URL https://doi.org/10.1007/s10851-014-0506-3.

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[7] Liam Carroll, Jesse Hoogland, Matthew Farrugia-Roberts, and Daniel Murfet. Dynamics of Transient Structure in In-Context Linear Regression Transformers, January 2025. arXiv:2501.17745.

[8] Bruno de Finetti. La prévision : ses lois logiques, ses sources subjectives. *Annales de l'Institut Henri Poincaré*, 7(1):1–68, 1937.

[9] Gregoire Deletang, Anian Ruoss, Paul-Ambroise Duquenne, Elliot Catt, Tim Genewein, Christopher Mattern, Jordi Grau-Moya, Li Kevin Wenliang, Matthew Aitchison, Laurent Orseau, Marcus Hutter, and Joel Veness. Language modeling is compression. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=jznbgiynus.

[10] Persi Diaconis and David Freedman. De Finetti's theorem for Markov chains. *The Annals of Probability*, 8(1):115–130, 1980.

[11] Ezra Edelman, Nikolaos Tsilivis, Benjamin L. Edelman, Eran Malach, and Surbhi Goel. The evolution of statistical induction heads: In-context learning markov chains. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=qaRT6QTIqJ.

[12] Fabian Falck, Ziyu Wang, and Christopher C. Holmes. Is in-context learning in large language models bayesian? a martingale perspective. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=b1YQ5WKY3w.

[13] Edwin Fong, Chris Holmes, and Stephen G Walker. Martingale posterior distributions. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 85(5):1357–1391, 2023. ISSN 1369-7412, 1467-9868. doi: 10.1093/jrsssb/qkad005.

[14] Sandra Fortini and Sonia Petrone. Quasi-Bayes properties of a procedure for sequential learning in mixture models. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 82(4):1087–1114, September 2020. ISSN 1369-7412. doi: 10.1111/rssb.12385.

[15] Sandra Fortini and Sonia Petrone. Exchangeability, Prediction and Predictive Modeling in Bayesian Statistics. *Statistical Science*, 40(1), January 2025. ISSN 0883-4237. doi: 10.1214/24-STS965.

[16] Sandra Fortini, Kenyon Ng, Sonia Petrone, Judith Rousseau, and Susan Wei. A principled framework for uncertainty decomposition in TabPFN, 2026. URL https://arxiv.org/abs/2602.04596v1. arXiv:2602.04596v1.

[17] Sandra Fortini, Kenyon Ng, Sonia Petrone, Judith Rousseau, and Susan Wei. Uncertainty decomposition for Bayes-filtered transformers via Bayesian predictive inference, 2026. URL https://arxiv.org/abs/2602.04596v2. arXiv:2602.04596v2.

[18] Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 30583–30598. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/c529dba08a146ea8d6cf715ae8930cbe-Paper-Conference.pdf.

[19] Tim Genewein, Grégoire Delétang, Anian Ruoss, Li Kevin Wenliang, Elliot Catt, Vincent Dutordoir, Jordi Grau-Moya, Laurent Orseau, Marcus Hutter, and Joel Veness. Memory-based meta-learning on non-stationary distributions. In *International Conference on Machine Learning (ICML)*, 2023.

[20] Jordi Grau-Moya, Tim Genewein, Marcus Hutter, Laurent Orseau, Gregoire Deletang, Elliot Catt, Anian Ruoss, Li Kevin Wenliang, Christopher Mattern, Matthew Aitchison, and Joel Veness. Learning universal predictors. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 16178–16205. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/grau-moya24a.html.

[21] Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A Transformer That Solves Small Tabular Classification Problems in a Second. In *International Conference on Learning Representations (ICLR)*, 2023.

[22] 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, January 2025. ISSN 1476-4687. doi: 10.1038/s41586-024-08328-6.

[23] Olav Kallenberg. *Foundations of Modern Probability*, volume 99 of *Probability Theory and Stochastic Modelling*. Springer, Cham, 3 edition, 2021.

[24] Soheil Kolouri, Kimia Nadjahi, Umut Simsekli, Roland Badeau, and Gustavo Rohde. Generalized sliced Wasserstein distances. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[25] Vladimir Mikulik, Grégoire Delétang, Tom McGrath, Tim Genewein, Miljan Martic, Shane Legg, and Pedro A. Ortega. Meta-trained agents implement Bayes-optimal agents. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[26] Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers Can Do Bayesian Inference. In *International Conference on Learning Representations (ICLR)*, 2022.

[27] Thomas Nagler and David Rügamer. Uncertainty quantification for prior-data fitted networks using martingale posteriors, 2025. arXiv:2505.11325.

[28] Neel Nanda and Joseph Bloom. Transformerlens. https://github.com/TransformerLensOrg/TransformerLens, 2022.

[29] Kenyon Ng, Edwin Fong, David Tyler Frazier, Jeremias Knoblauch, and Susan Wei. TabMGP: Martingale posterior with TabPFN. In Tong Zhang, Miroslav Dudik, Martin Jaggi, Alekh Agarwal, Sharon Li, Dale Schuurmans, Jerry Zhu, Felix Berkenkamp, Hanze Dong, and Alberto Bietti, editors, *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 92242–92277. PMLR, 06–11 Jul 2026. URL https://proceedings.mlr.press/v306/ng26a.html.

[30] Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads. *Transformer Circuits Thread*, 2022. https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

[31] Pedro A. Ortega, Jane X. Wang, Mark Rowland, Tim Genewein, Zeb Kurth-Nelson, Razvan Pascanu, Nicolas Heess, Joel Veness, Alex Pritzel, Pablo Sprechmann, Siddhant M. Jayakumar, Tom McGrath, Kevin Miller, Mohammad Azar, Ian Osband, Neil Rabinowitz, András György, Silvia Chiappa, Simon Osindero, Yee Whye Teh, Hado van Hasselt, Nando de Freitas, Matthew Botvinick, and Shane Legg. Meta-learning of Sequential Strategies, July 2019. arXiv:1905.03030.

[32] Madhur Panwar, Kabir Ahuja, and Navin Goyal. In-context learning through the bayesian prism. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=HX5ujdsSon.

[33] Core Francisco Park, Ekdeep Singh Lubana, Itamar Pres, and Hidenori Tanaka. Competition Dynamics Shape Algorithmic Phases of In-Context Learning. In *International Conference on Learning Representations (ICLR)*, 2025. Spotlight.

[34] Jingang QU, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICL: A tabular foundation model for in-context learning on large data. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=0VvD1PmNzM.

[35] Allan Raventós, Mansheej Paul, Feng Chen, and Surya Ganguli. Pretraining task diversity and the emergence of non-Bayesian in-context learning for regression. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[36] Maria L. Rizzo and Gábor J. Székely. Energy distance. *WIREs Computational Statistics*, 8(1): 27–38, January 2016. ISSN 1939-5108, 1939-0068. doi: 10.1002/wics.1375.

[37] Adam S. Shai, Sarah E. Marzen, Lucas Teixeira, Alexander Gietelink Oldenziel, and Paul M. Riechers. Transformers represent belief state geometry in their residual stream. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 75012–75034. Curran Associates, Inc., 2024. doi: 10.52202/079017-2387. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/8936fa1691764912d9519e1b5673ea66-Paper-Conference.pdf.

[38] Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, Joao Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 35151–35174. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/von-oswald23a.html.

[39] Miles Wang, Tom Dupre la Tour, Olivia Watkins, Aleksandar Makelov, Ryan Andrew Chi, Samuel Miserendino, Jeffrey George Wang, Achyuta Rajaram, Johannes Heidecke, Tejal Patwardhan, and Daniel P Mossing. Persona features control emergent misalignment. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=yjrVOxjkDR.

[40] Jingfeng Wu, Difan Zou, Zixiang Chen, Vladimir Braverman, Quanquan Gu, and Peter Bartlett. How many pretraining tasks are needed for in-context learning of linear regression? In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=vSh5ePa0ph.

[41] Daniel Wurgaft, Ekdeep S Lubana, Core Francisco Park, Hidenori Tanaka, Gautam Reddy, and Noah Goodman. In-context learning strategies emerge rationally. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen, editors, *Advances in Neural Information Processing Systems*, volume 38, pages 118425–118485. Curran Associates, Inc., 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/abb22b1e4b4ab2988862be9d7ed083f9-Paper-Conference.pdf.

[42] Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An Explanation of In-context Learning as Implicit Bayesian Inference. In *International Conference on Learning Representations (ICLR)*, 2022.

[43] Ruiqi Zhang, Spencer Frei, and Peter L. Bartlett. Trained transformers learn linear models in-context. *Journal of Machine Learning Research*, 25(49):1–55, 2024. URL http://jmlr.org/papers/v25/23-1042.html.

Appendix contents

A Notation summary 16
B Beta-Bernoulli 16
C General k-Markov exchangeability 17
D Sufficient conditions for asymptotic exchangeability 18
E Reliability of PMC estimates 19
F Experimental setup 25
G Linear regression 29
H Balls-and-urns 35
I Markov chains 38
J Marginal densities and CDFs from PMC 41

A Notation summary

Table 2 records the main notation used throughout the manuscript.

Table 2: Core notation used throughout the paper.

Symbol	Meaning
$\mathcal{Y}$	observation space: finite for balls-and-urns and Markov chains, $\mathbb{R}$ for the linear-regression response
$\theta \in \Theta$	latent task parameter and its space (e.g. regression coefficient w, urn probability vector p, transition matrix Q)
$\tilde{\theta}$	the latent task as a random variable, $\tilde{\theta} \sim \Pi$; plain θ denotes values
$\tilde{P}$	directing random measure of an exchangeable sequence (Section 2.1)
$\tilde{Q}$	random transition matrix of a 1-Markov exchangeable sequence (Section 2.2)
$Q_{a,b}$, $Q_{a,\cdot}$	entry and row of a transition matrix $Q \in \mathcal{Q}$
$N_{a,b}(y_{1:n})$	number of transitions from a to b in $y_{1:n}$ (Section 2.2)
Π	prior on the latent task space (instantiated as Π_∞ or Π_M)
Π_∞	population prior on the latent task space
Π_M	empirical prior formed from M sampled tasks
M	task diversity, i.e. number of latent tasks in the empirical prior
P_θ	probability kernel from $\mathcal{Y}^*$ to $\mathcal{Y}$: after history h, the next token is drawn from $P_\theta(\cdot \mid h)$
$\mathbb{P}$	joint law (2) on $\mathcal{Y}^\infty$ with prior Π
$\mathbb{P}_M$	joint law (2) on $\mathcal{Y}^\infty$ with prior Π_M
$\mathbb{P}_\infty$	joint law (2) on $\mathcal{Y}^\infty$ with prior Π_∞
T_ϕ	sequence model $\mathcal{Y}^* \to \Delta(\mathcal{Y})$, e.g. a transformer
$\mathbb{P}_\phi$	joint law on $\mathcal{Y}^\infty$ with one-step predictives T_ϕ
$\tilde{P}_\phi$	directing random measure of (Y_n) under $\mathbb{P}_\phi$, or of the exchangeable limit of its shifts (Section 4)
$\tilde{Q}_\phi$	random transition matrix of (Y_n) under $\mathbb{P}_\phi$ (Section 4)
$\mathcal{P}$	set of probability measures on $\mathcal{Y}$
$\theta(\cdot)$	functional on $\mathcal{P}$ or $\mathcal{Q}$ with values in $\mathbb{R}^D$; $\theta(y_{1:\infty})$ is its value computed from an infinite path (Section 4)
P_n	one-step predictive of $\mathbb{P}$; $(P_n)_{n\geq 0}$ is the predictive rule, see (1)
n	pretraining sequence length
c	evaluation prompt, $c = y_{1:\ell}$
ℓ	evaluation prompt length
N	PMC rollout length
R	number of independent rollouts in PMC
$\Pi^{\mathrm{mem}}(\cdot \mid c)$	memorizing posterior, under the prior Π_M, see (7)
$\Pi^{\mathrm{gen}}(\cdot \mid c)$	generalizing posterior, under the prior Π_∞, see (7)
$\widehat{\Pi}(\cdot \mid c)$	PMC posterior, the empirical law of the R PMC samples given prompt c; $\widehat{\Pi}(\cdot \mid \varnothing)$ is the PMC prior

B Beta-Bernoulli

Pretraining distribution. Training sequences were drawn by first drawing a probability $\tilde{\theta}$ from a $\mathrm{Beta}(1, 1)$ prior, then drawing independent Bernoulli observations with probability $\tilde{\theta}$.

Transformer architecture. The Beta-Bernoulli BFT was trained with 2 layers with embedding dimension $d_{\mathrm{model}} = 128$, 2 attention heads with dimension $d_{\mathrm{head}} = 32$, MLP hidden dimension $d_{\mathrm{mlp}} = 512$ and a context length of 512 tokens (a BOS token followed by 511 observations). The BOS token allows the model to represent an unconditioned predictive distribution.

Training details. Training was performed over $100{,}000$ steps with a batch size of 128 and seed 0. We use a linear learning rate schedule with 500 warmup steps, staying constant at 10^{-4} thereafter. The model is trained autoregressively with next-token cross-entropy loss.

PMC. For a prompt $y_{1:\ell}$, let $S = \sum_{i=1}^{\ell} y_i$. The conjugate posterior is

$$\tilde{\theta} \mid y_{1:\ell} \sim \mathrm{Beta}(1 + S, 1 + \ell - S).$$

The rollout length is $N = 450$, and every panel uses $R = 1000$ rollouts. Figure 1(b) is the $\theta^\star = 0.2$ panel of the grid in Figure 5, shown on its own for the main text. In Figure 5, the PMC prior and posterior over θ are compared with the $\mathrm{Beta}(1, 1)$ prior and the conjugate posterior. Each panel conditions on a different prompt $y_{1:\ell}$ of length $\ell = 32$. Reading the panels left-to-right, top-to-bottom, the underlying ground truth probability takes values $\theta^\star = 0.1, 0.2, \ldots, 0.9$.

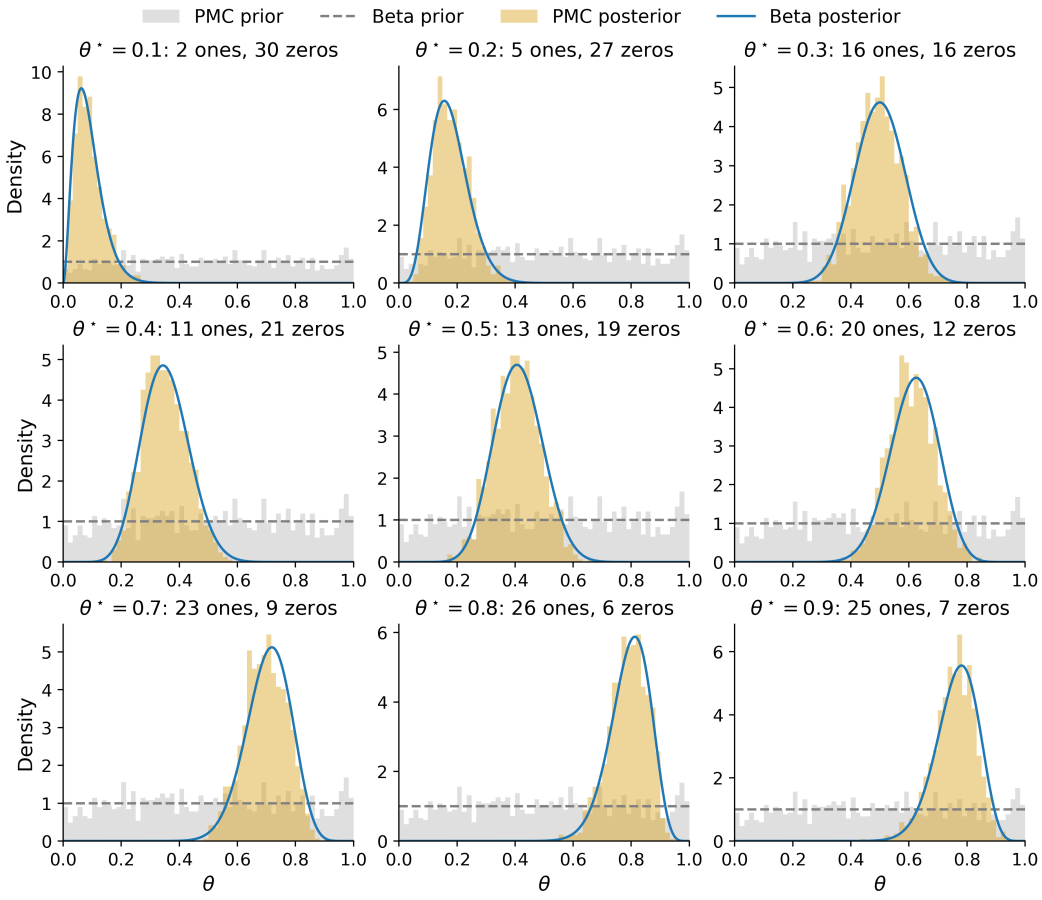


Figure 5: **Beta-Bernoulli: PMC prior and posterior across prompts.** PMC prior and posterior over θ versus the analytic $\mathrm{Beta}(1, 1)$ prior and conjugate posterior.

C General k-Markov exchangeability

For completeness, we record the generalization of Section 2.2 to higher Markov orders. Our experiments use only $k = 0$ and $k = 1$, but the higher-order case covers the $(k + 1)$-gram models of Edelman et al. [11], in which the next-token distribution is drawn afresh for each length-k context.

Fix $k \geq 1$. We call (Y_i) *k-Markov exchangeable* if the sequence of overlapping k-blocks

$$(Y_{1:k}),\ (Y_{2:k+1}),\ (Y_{3:k+2}),\ \ldots$$

is Markov exchangeable in the sense of Section 2.2 when viewed as a sequence on $\mathcal{Y}^k$.

Under the corresponding recurrence condition, the Diaconis–Freedman representation yields a mixture of 1-step Markov chains on $\mathcal{Y}^k$. Because consecutive blocks of a path overlap in $k-1$ coordinates, the mixing kernel is supported on transitions between overlapping blocks, and a 1-step chain on $\mathcal{Y}^k$ with that support is exactly a kth-order Markov chain on $\mathcal{Y}$.

For finite $\mathcal{Y}$, a kth-order transition matrix is a row-stochastic array

$$Q = (Q_{u,y})_{u\in\mathcal{Y}^k,\, y\in\mathcal{Y}}, \qquad \sum_{y\in\mathcal{Y}} Q_{u,y} = 1 \quad \text{for every } u \in \mathcal{Y}^k.$$

Writing $\mathcal{Q}_k$ for the set of such arrays, (3) holds with Π supported on $\mathcal{Q}_k$: the latent kernel depends on the history only through the last k tokens. A population prior on $\mathcal{Q}_k$ is typically implemented as independent Dirichlet priors over the transition probabilities indexed by length-k contexts.

D Sufficient conditions for asymptotic exchangeability

PMC is applied to the trained BFT T_ϕ, so the conditions below are conditions on the joint law $\mathbb{P}_\phi$ (Section 3).

Following Fortini and Petrone [15], $(Y_n)_{n\geq 1}$ is asymptotically exchangeable under $\mathbb{P}_\phi$ with limit directing random measure $\tilde{P}_\phi$ if, as $n\to\infty$, $(Y_{n+1}, Y_{n+2}, \dots)$ converges in distribution to an exchangeable sequence with directing random measure $\tilde{P}_\phi$. If the predictives $T_\phi(\cdot \mid Y_{1:n})$ converge weakly, $\mathbb{P}_\phi$-almost surely, to a random probability measure $\tilde{P}_\phi$, then $(Y_n)_{n\geq 1}$ is asymptotically exchangeable with limit directing random measure $\tilde{P}_\phi$ [2, 15]. The empirical distributions of $Y_{1:n}$ also converge weakly to $\tilde{P}_\phi$, $\mathbb{P}_\phi$-almost surely. This follows from the strong law of large numbers for the bounded martingale differences $f(Y_n) - \int f \,\mathrm{d}T_\phi(\cdot \mid Y_{1:n-1})$, for bounded f. For linear regression, $\theta(\tilde{P}_\phi)$ involves $\mathbb{E}_{\tilde{P}_\phi}[XY]$, and we assume in addition that $\frac{1}{n}\sum_{i=1}^n X_i Y_i$ converges to $\mathbb{E}_{\tilde{P}_\phi}[XY]$, $\mathbb{P}_\phi$-almost surely.

The BPI literature provides several sufficient conditions on $\mathbb{P}_\phi$ for the convergence of the predictives. Condition (i) implies (ii). Condition (ii) implies (iii) and (iv), because a martingale has zero conditional drift.

(i) $\mathbb{P}_\phi$ is exchangeable. The predictive distributions of an exchangeable sequence converge weakly, almost surely, to its directing random measure [15], so $T_\phi(\cdot \mid Y_{1:n}) \Rightarrow \tilde{P}$, $\mathbb{P}_\phi$-almost surely.

(ii) $(T_\phi(\cdot \mid Y_{1:n}))_{n\geq 0}$ is a measure-valued martingale, equivalently $\mathbb{P}_\phi$ is conditionally identically distributed (c.i.d.); martingale convergence then supplies the almost-sure limit [15]. The martingale property has been checked empirically on trained models. Falck et al. [12] show it fails for in-context predictives of large language models. Nagler and Rügamer [27] find that TabPFN fails it as well, and for that reason they abandon TabPFN as the predictive rule after the first step, switching to a copula update that preserves the martingale property.

(iii) $\mathbb{P}_\phi$ is almost conditionally identically distributed (a.c.i.d.) [2]. That is, there are nonnegative random variables ξ_n, each a function of $Y_{1:n}$, with $\sum_n \xi_n < \infty$ almost surely, such that $\mathrm{TV}\big(\mathbb{P}_\phi(Y_{n+2} \in \cdot \mid Y_{1:n}), \mathbb{P}_\phi(Y_{n+1} \in \cdot \mid Y_{1:n})\big) \leq \xi_n$ almost surely for every n. Ng et al. [29] check this condition empirically for TabPFN and find that TabPFN departs from it, yet the resulting martingale posteriors remain stable with near-nominal frequentist coverage, suggesting the condition is stricter than necessary.

(iv) $(T_\phi(\cdot \mid Y_{1:n}))_{n\geq 0}$ is a quasi-martingale, setwise: the predictive update $\Delta_n(A) := T_\phi(A \mid Y_{1:n}) - T_\phi(A \mid Y_{1:n-1})$ satisfies, for every $A \subseteq \mathcal{Y}$,

$$\sum_{n\geq 1} \mathbb{E}_{\mathbb{P}_\phi}\big[\big|\mathbb{E}_{\mathbb{P}_\phi}[\Delta_n(A) \mid Y_{1:n-1}]\big|\big] < \infty. \tag{8}$$

Fortini et al. [16] check this condition empirically for TabPFN; the Monte Carlo budgets required are so large that the verification has so far been inconclusive. A subsequent version [17] relaxes this quasi-martingale condition to a weaker *signed* condition, requiring only that the signed tail sums of the conditional drift vanish, and reports positive diagnostics on a small Beta-Bernoulli BFT while the TabPFN diagnostics remain inconclusive.

Verification in this paper. We do not verify any of these conditions for the BFTs studied here, either theoretically or empirically. As a proxy, we plot the finite-rollout estimates along each rollout against the rollout length (the "jellyfish" plots, Figures 7, 8, and 9) and check whether they stabilize. In the controlled families of Section 5 the baseline priors and posteriors over θ are available in closed form, so the PMC samples can be compared against them directly. At low task diversity, the PMC prior matches the training prior Π_M and, on in-distribution prompts, the PMC posterior matches the memorizing posterior $\Pi^{\mathrm{mem}}(\cdot \mid c)$. At high task diversity, the PMC prior matches the population prior Π_∞ and the PMC posterior matches the generalizing posterior $\Pi^{\mathrm{gen}}(\cdot \mid c)$. Agreement between the PMC samples and a baseline does not show that the sufficient conditions hold. Appendix E examines how the distance metrics between the PMC samples and the baselines depend on the number of rollouts, the latent task dimension, and the rollout length.

E Reliability of PMC Estimates

We examine how number of rollouts R, latent task dimension, and rollout length N affect the distances between PMC samples and the memorizing and generalizing baselines.

We compare PMC samples with samples from the memorizing and generalizing baselines using energy distance (11) and sliced Wasserstein-1 (12), with the estimators specified in Appendix F. We call these distances *recovery error*, although neither baseline is known to be the BFT's true implicit prior or posterior. The measured discrepancy can reflect a mismatch between the BFT's implicit prior or posterior and the baselines, errors from computing the estimates on a finite rollout, and Monte Carlo error.

Settings. Table 3 summarizes the default experimental settings. Linear-regression and balls-and-urns models use no positional encoding, while 1-Markov models use RoPE. Training sequences contain 256 (x, y) pairs for linear regression and 256 tokens for balls-and-urns and 1-Markov. Each posterior PMC run uses one in-distribution prompt.

Table 3: Default settings of the PMC runs in Appendices E.2–E.5.

Setting	Value
Posterior prompt length ℓ	32 observations
Number of rollouts R	1024
Rollout length N	500
Sliced Wasserstein projections	100
Baseline samples	1024 per baseline and source, held fixed within each run
Posterior prompts	One in-distribution prompt
Seed	42

Each posterior prompt contains 32 observations. We evaluate all prior and posterior runs up to $N = 500$, beyond their training context lengths. For 1-Markov, we rescale RoPE positions by $256/544$ to remain within the training range (rotary position interpolation). The 1-Markov runs in Appendices E.3–E.5 use the same rescaling. The experiments involving 1-Markov BFTs in the main text use the original RoPE positions, as we do not roll out past the training context length. Rescaling changes the model's predictions at every rollout length.

For balls-and-urns and linear regression, as we train without positional encodings, we can continue to rollout and evaluate past the BFT's training context length without modifying its weights. Rolling out further than the training context length still represents a distribution shift for the BFT, as it has to form predictions on positions it has not seen before. It is not guaranteed that the BFT's predictions on positions out of its training context length will be accurate; we check the accuracy of balls-and-urns BFTs and 1-Markov BFTs on sequences longer than the training context length in Appendix E.1.

E.1 Agreement of next-token predictives with exact PPDs

For balls-and-urns and 1-Markov, we compare the model's next-token distribution with the exact predictive using symmetrized KL. We evaluate 256 in-distribution sequences of length 500 against the

memorizing predictive and 256 out-of-distribution sequences of length 500 against the generalizing predictive. We report averages across sequences at each position (Figure 6) and across all positions. We omit linear regression because its baseline predictive is a continuous density, while the BFT's output is binned.

For balls-and-urns at $M = 32$, the symmetrized KL averaged over sequences and positions is 4.8×10^{-4} on in-distribution sequences and 0.244 on out-of-distribution sequences. The in-distribution value stays near 3.4×10^{-4} from position 128 through position 500, beyond the training context. For 1-Markov, the symmetrized KL averaged over sequences and positions is 0.100 on in-distribution sequences and 0.284 on out-of-distribution sequences.

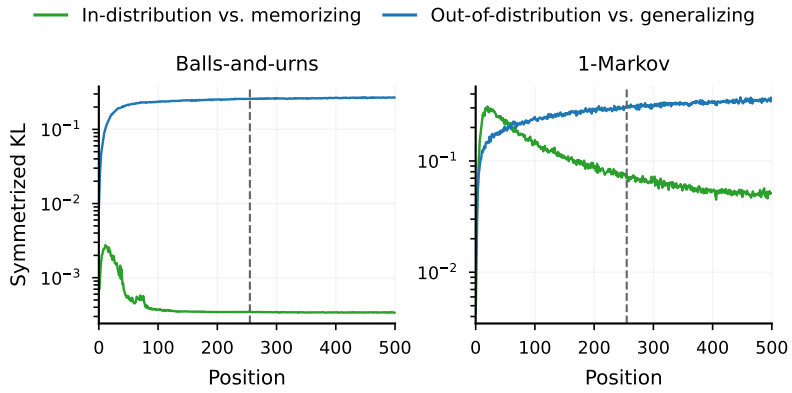


Figure 6: **Next-token predictions compared with Bayesian baselines.** Both models were trained at task diversity $M = 32$. At each position, we average the symmetrized KL between the model's next-token distribution and the baseline predictive over 256 sequences. In-distribution sequences are evaluated against the memorizing predictive (green), and out-of-distribution sequences against the generalizing predictive (blue). Dashed lines mark the training-context boundary. Both vertical axes are logarithmic and use different ranges. The 1-Markov model uses the position rescaling described in Appendix E.

E.2 Path stability along a rollout

We assess the stability of the latent-task estimate along rollouts from the BFT's predictive distributions. Under the assumptions of Appendix D, this estimate converges almost surely. Let $\hat{\theta}_\ell$ be the latent-task estimate computed from the observed prompt and $\hat{\theta}_{\ell+N}$ the estimate after appending N generated observations. For each prompt, we average

$$\frac{1}{D}\|\hat{\theta}_{\ell+N} - \hat{\theta}_\ell\|_1$$

over rollouts and plot the result against N. D is the dimension of $\hat{\theta}$: 8 for linear regression, 12 for balls-and-urns, and 100 for 1-Markov. Figures 7–9 show the results at $M = 32$. For each task family, we use 20 in-distribution prompts, each containing 32 observations, and generate 128 rollouts per prompt. Each curve corresponds to one prompt. The curves being flat is a sign the estimates are stabilizing, but this does not show the sufficient conditions hold.

E.3 Recovery error versus number of rollouts

To examine the effect of the number of rollouts, we fix the rollout length at $N = 500$ and vary $R \in \{16, 32, 64, 128, 256, 512, 1024\}$. For every value of R, we independently generate 20 sets of R rollouts. We draw 1024 latent-parameter samples from each baseline distribution and keep these samples fixed across all rollout counts and repeats. For each metric and baseline, we compute one distance per rollout set, giving 20 distance estimates at each R. The curves show the mean of these estimates, and the error bars show their standard deviation. Figures 10 and 11 show the posterior and prior results, respectively.

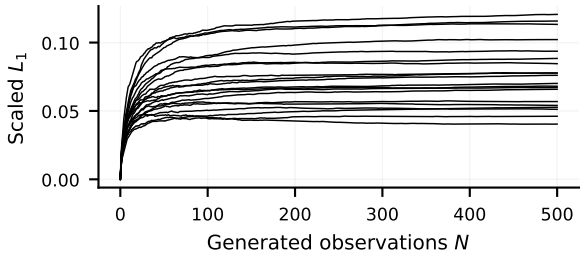


Figure 7: **Path stability along a rollout, linear regression.** Mean scaled L_1 distance from the estimate computed from the observed prompt. Each curve represents one prompt and averages 128 rollouts. Rollouts extend to $N = 500$.

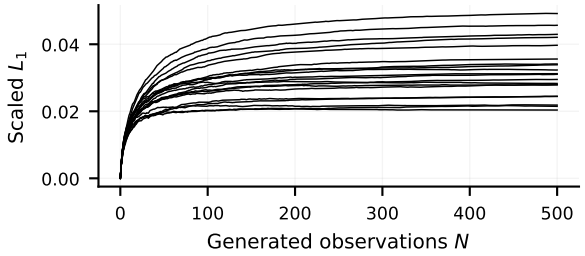


Figure 8: **Path stability along a rollout, balls-and-urns.** Axes, curves and settings follow Figure 7.

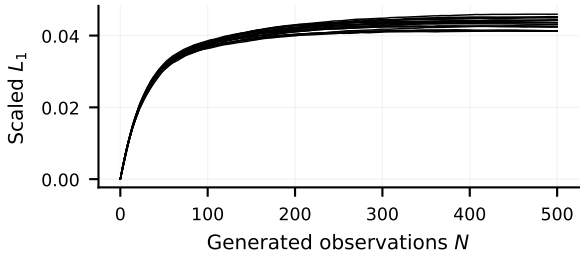


Figure 9: **Path stability along a rollout,** 1**-Markov.** Axes and curves follow Figure 7. RoPE positions use the rescaling described in Appendix E.

Increasing the number of rollouts generally reduces variation across repeats. For linear regression, the mean posterior energy distance to the memorizing baseline remains near 0.0564, while its standard deviation decreases from 0.0063 at $R = 16$ to 0.00073 at $R = 512$, with no further decrease at $R = 1024$. The error bars measure how much the estimated distance changes when the model generates a new set of rollouts. Throughout these repeats, the prompt, the samples drawn from each baseline, and the projection directions remain fixed. The error bars therefore describe variation caused by sampling the PMC rollouts. They do not show how the results would change with a different prompt, different samples from the baselines, or different projection directions.

For 1-Markov, sliced Wasserstein distance places the PMC posterior closer to the generalizing baseline at every tested number of rollouts. At $R = 16$, the distances to the memorizing and generalizing baselines are 0.0774 and 0.0432, respectively. At $R = 1024$, they are 0.0772 and 0.0396. Both distances change little as the number of rollouts increases. Because these results concern a single prompt, evaluation on additional prompts is needed to determine whether the PMC posterior consistently remains closer to the generalizing baseline.

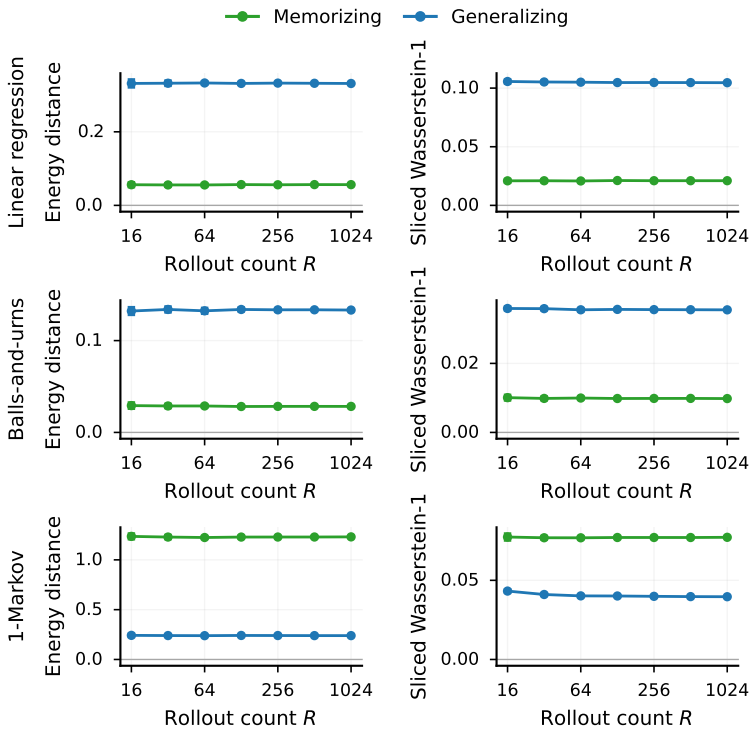


Figure 10: **Posterior recovery error versus number of rollouts**, $M = 32$. Rows show linear regression, balls-and-urns, and 1-Markov. Columns show energy distance and sliced Wasserstein-1. Green denotes the memorizing baseline, and blue denotes the generalizing baseline. All three families use $N = 500$. At each value of R, solid curves show the mean distance across 20 independently generated sets of rollouts, and error bars show the standard deviation across these sets.

E.4 Recovery error versus latent task dimension

We vary the alphabet size at $M = 32$, using balls-and-urns BFTs with $|\mathcal{Y}| \in \{6, 12, 24, 48\}$ and 1-Markov BFTs with $|\mathcal{Y}| \in \{3, 4, 6, 8\}$. Their latent tasks have, respectively, $|\mathcal{Y}| - 1 \in \{5, 11, 23, 47\}$ and $|\mathcal{Y}|(|\mathcal{Y}| - 1) \in \{6, 12, 30, 56\}$ free parameters. Each alphabet size $|\mathcal{Y}|$ corresponds to a newly trained model with its own task distribution Π_M, except for $|\mathcal{Y}| = 12$ in the balls-and-urns setting where we use the model trained for the main experiment. All models are trained on a single seed. Linear regression is not included in this experiment. The architecture and training procedure follow Appendix F. Figures 12 and 13 report results at $N = 500$ and $R = 1024$. The 1-Markov runs use the position rescaling described in Appendix E.

For the balls-and-urns posterior, both metrics favor the memorizing baseline at every tested dimension. For the 1-Markov posterior, both metrics favor the memorizing baseline at dimension 6 and the generalizing baseline at dimensions 30 and 56; they disagree at dimension 12. The 1-Markov BFT in the main text has 90 free parameters, beyond the largest dimension tested here.

E.5 Recovery error versus rollout length

Figures 14 and 15 show the posterior and prior comparisons as rollout length increases, at $M = 32$. Rollouts extend to $N = 500$ for all three families, evaluated at $N \in \{8, 16, 32, 64, 128, 223, 256, 384, 500\}$; 1-Markov reaches the lengths beyond 223 through the position rescaling of Appendix E.

For the linear-regression prior (Figure 15), both metrics place the PMC samples closer to the memorizing baseline from $N = 16$ onward, and the gap widens as N grows. At $N = 8$, linear regression fits 8 coefficients to only 8 generated (x, y) pairs, so small changes in the generated

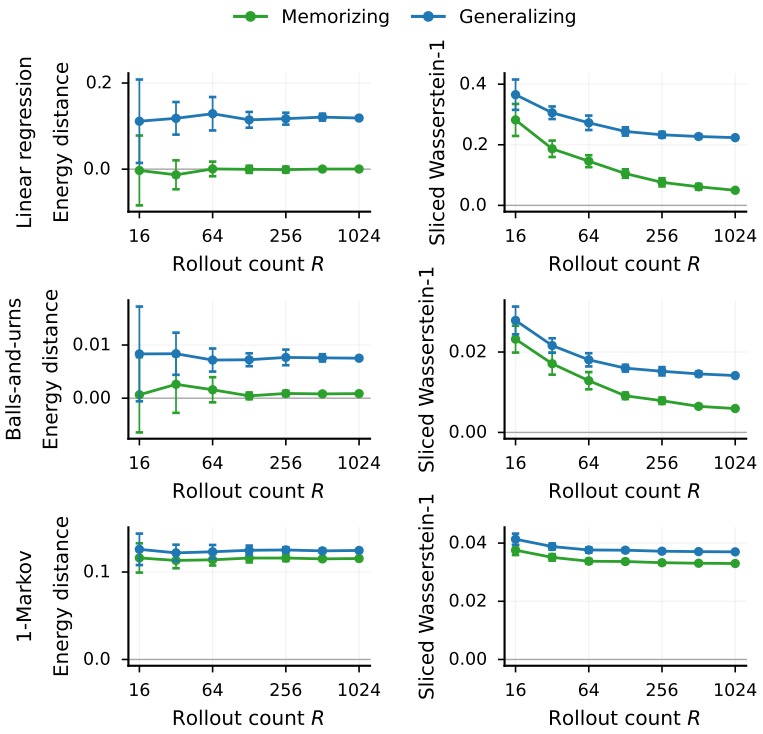


Figure 11: **Prior recovery error versus number of rollouts**, with settings and visual conventions as in Figure 10. Negative unbiased energy-distance values are retained.

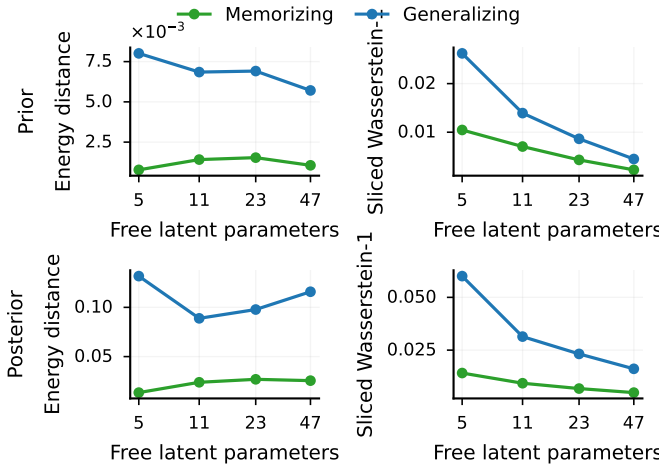


Figure 12: **Balls-and-urns recovery error versus latent task dimension**, $M = 32$, $N = 500$, $R = 1024$. The horizontal axis counts free parameters $|\mathcal{Y}| - 1 \in \{5, 11, 23, 47\}$. Rows: prior and posterior; columns: energy distance and sliced Wasserstein-1. Green and blue denote the memorizing and generalizing baselines.

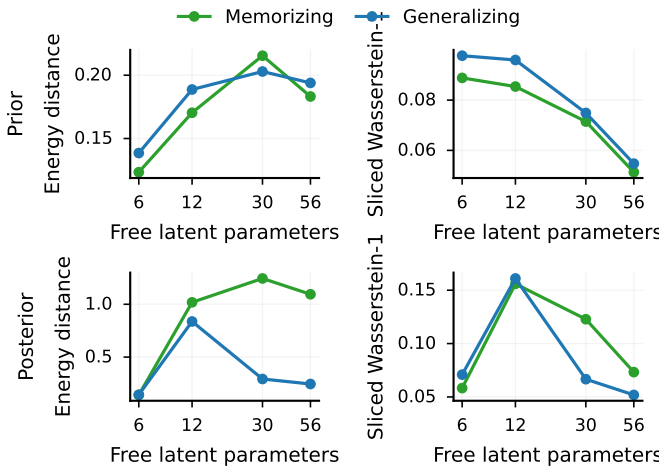


Figure 13: **Markov recovery error versus latent task dimension**, $M = 32$, $N = 500$, $R = 1024$. The state-count sweep stops at $|\mathcal{Y}| = 8$, or 56 free transition parameters, below the main experiment's 90. Rows, columns, and colors follow Figure 12.

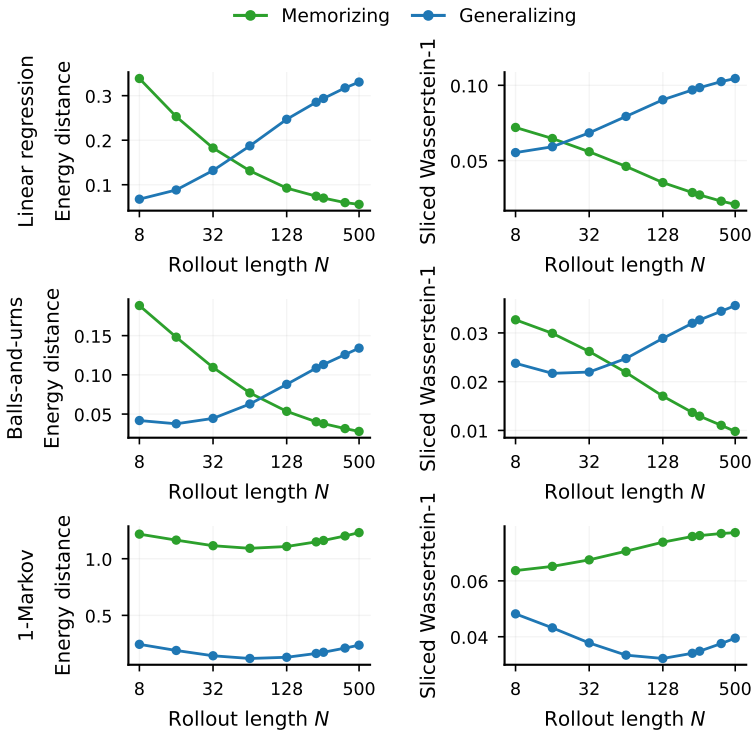


Figure 14: **Posterior recovery error versus rollout length**, $M = 32$, $R = 1024$. Rows show linear regression, balls-and-urns, and 1-Markov. Columns show energy distance and sliced Wasserstein-1. Green denotes the memorizing baseline, and blue denotes the generalizing baseline. The 1-Markov model uses the position rescaling described in Appendix E.

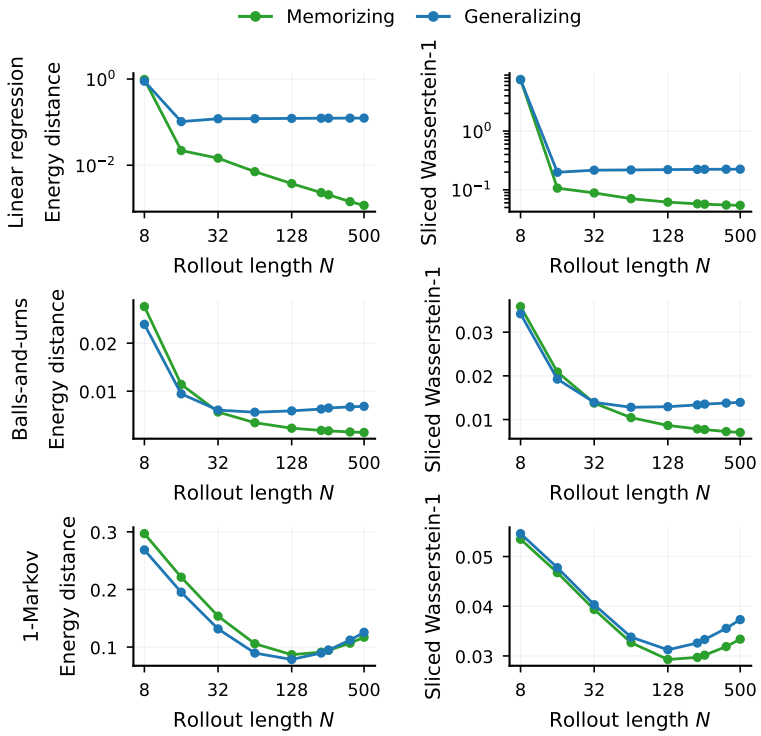


Figure 15: **Prior recovery error versus rollout length**, $M = 32$, $R = 1024$. Rows, columns, and colors follow Figure 14. The memorizing baseline is the training prior Π_M, and the generalizing baseline is the population prior Π_∞. Logarithmic vertical axes are used where distances span several orders of magnitude.

observations can lead to large changes in the estimated coefficients, and this causes a large discrepancy between the PMC samples and the samples from the baselines.

For the balls-and-urns prior, both metrics place the PMC samples closer to the memorizing baseline from $N = 32$ onward, and the difference between the distances to the two baselines widens as rollout length increases. For the 1-Markov prior, the metrics identify different baselines as closer within the training context: at $N = 223$ energy distance gives a smaller distance to the generalizing baseline, whereas sliced Wasserstein gives a smaller distance to the memorizing baseline. The two agree from $N = 256$ onward, where energy distance also favors the memorizing baseline.

For posterior PMC (Figure 14), which baseline is closer also depends on rollout length. At $N = 8$, the PMC samples are closer to the generalizing baseline under both metrics in all three families. In linear regression, energy distance first places the samples closer to the memorizing baseline at $N = 64$ among the lengths tested. Sliced Wasserstein does so from $N = 32$. For balls-and-urns, the memorizing baseline first becomes closer at $N = 128$ under energy distance and at $N = 64$ under sliced Wasserstein. For 1-Markov, neither metric places the samples closer to the memorizing baseline at any tested length; both favor the generalizing baseline throughout.

F Experimental setup

We record the training protocol, architectural choices, and reporting conventions shared across task families. Family-specific instantiations (state-space size, model width, PMC rollout length, task-diversity grid) are in Appendices G, H, and I.

Training protocol. For each task family f and each chosen task-diversity level M, we train a separate transformer on sequences drawn from the empirical mixture law $\mathbb{P}_M$, equivalently from the empirical task prior Π_M. We write n for the meta-training sequence length and use one fixed value of n per

task family in the main text. At evaluation time we draw prompts from the length-ℓ marginals of $\mathbb{P}_M$ and $\mathbb{P}$. Prompt length ℓ may vary within a family, and checkpoint sweeps may be included when studying transient behavior, but neither prompt source nor checkpoint index is treated as a primary axis of the experimental design.

Architecture. The architectural choices of the cited papers for each task family, and our own, are listed in Table 4. Entries marked with † are not explicitly stated in the cited paper but inferred from the named architecture (GPT-2 implies learned absolute positional embeddings, and GPT-NeoX implies RoPE). The main difference from Raventós et al. [35] and Wurgaft et al. [41] in linear regression is the output head: where Raventós et al. [35] and Wurgaft et al. [41] use a scalar output head trained under squared-error loss, we discretize the response variable and use a categorical output head trained under log loss, so that the transformer returns the full next-token distribution needed for PMC rollouts. Squared-error training coincides with the meta-learning objective (6) under a Gaussian output head of fixed variance and therefore fits only the mean of the PPD. Using a single categorical head across all three families keeps the architecture and loss uniform, at the price of not exploiting the closed form of the linear regression PPD. For linear regression, we use a different attention masking scheme than causal masking to enforce covariates to attend to covariate-response pairs. We refer to this masking scheme as 'Autoregressive-PFN' and refer to Appendix G for details.

For the two discrete families (balls-and-urns and Markov chains), a BOS token is required so that the BFT can express a marginal predictive $T_\phi(\cdot \mid \varnothing)$ for the first observation, matching P_0 in the ideal limit. For linear regression, the exogenous covariate x_1 seeds the first prediction and no BOS token is needed. We refer to Appendices G, H, and I for particular details on the transformer architecture used in each task family. We include a summary of training hyperparameters in the main text in Table 5.

Table 4: **Transformer architecture of the cited papers and of ours, by task family.** Entries labeled with † are inferred from the named architecture and not explicitly stated in the text. See Table 5 for specific task-family training hyperparameters.

Task family	Source	Attention	Pos. encoding	Output head	Sequence length	BOS
Linear regression	Raventós et al. [35]	Causal	Learned absolute†	Scalar (MSE)	16	No
	Wurgaft et al. [41]	Causal†	RoPE†	Scalar (MSE)	{16, 32, 64}	No
	Ours	Autoregressive-PFN	None (NoPE)	Categorical (log loss)	256	No
Balls-and-urns	Wurgaft et al. [41]	Causal†	RoPE†	Categorical (log loss)	{128, 256, 320}	N/A
	Ours	Causal	None (NoPE)	Categorical (log loss)	256	Yes
Markov chains (row-wise Dirichlet)	Park et al. [33]	Causal	RoPE	Categorical (log loss)	512	No
	Ours	Causal	RoPE	Categorical (log loss)	256	Yes

Table 5: **Training hyperparameters.**

Category	Hyperparameter	Linear regression	Balls-and-urns	Markov chains
Data	Covariate dimension (d)	8	N/A	N/A
	State-space size ($\lvert\mathcal{Y}\rvert$)	N/A	12	10
	Observation noise variance (σ^2)	0.25	N/A	N/A
	Training sequence length	256 pairs	255 observations (+ BOS)	256 observations (+ BOS)
Model	Number of layers	2	2	2
	Number of attention heads	4	1	4
	Embedding dimension	512	64	64
	Attention head dimension	128	64	16
	MLP hidden dimension	512	256	256
	Activation	GELU	GELU	ReLU
	Layer normalization	Pre	Pre	Pre
	Positional encoding	None (NoPE)	None (NoPE)	RoPE
	Attention masking	Autoregressive-PFN	Causal	Causal
	Output head	Density on $\mathbb{R}$, 256 bins on $[-10, 10]$, half-normal tails	Categorical, vocabulary 12	Categorical, vocabulary 11
	BOS token	No	Yes	Yes
	Max context length	512	256	256
	Total parameters	$\approx$ 3.5M	$\approx$ 118K	$\approx$ 100K
Optimizer	Type	AdamW	AdamW	AdamW
	Weight decay	10^{-5}	10^{-5}	0.01
	Gradient clipping (norm)	1	1	N/A
LR schedule	Shape	Linear warmup, then constant	Constant (no warmup)	Constant (no warmup)
	Peak learning rate	10^{-4}	5×10^{-4}	6×10^{-4}
	Warmup steps	1,000	0	0
Experiments	Batch size	256	64	128
	Training steps	150,000	100,000	100,000
	Random seed	42	42	42
	Task-diversity grid	$\{2^j : 0 \le j \le 16\}$	$\{2^j : 0 \le j \le 12\}$	$\{2^j : 2 \le j \le 11\}$

PMC settings. Table 6 collects the PMC settings used for the experiments in the main text; the appendices give specific implementation details for each task family. The reliability study of Appendix E uses its own settings, recorded separately in Table 3.

Table 6: **PMC settings across task families.** Settings used for the task-diversity sweep and the transient-generalization experiment. ℓ is the evaluation prompt length, N the rollout length, and R the number of rollouts. Prior rollouts condition on the BOS token alone for balls-and-urns and Markov chains, and on a single covariate $x_1 \sim \mathcal{N}(0, I_d)$ for linear regression.

Setting	Linear regression	Balls-and-urns	Markov chains
Prompt length ℓ	32	32	32
Rollout length N	223	223	223
Posterior rollouts R per prompt	100	100	100
Prior rollouts R	1024	1024	1024
Evaluation prompts per source	128	128	128
In-distribution prompt source	$w \sim \Pi_M$	$p \sim \Pi_M$	Q from the training pool
Out-of-distribution prompt source	$w \sim \mathcal{N}(0, I_d)$	$p \sim \mathrm{Dirichlet}(1, \dots, 1)$	rows $\sim \mathrm{Dirichlet}(1, \dots, 1)$
Rollout statistic	OLS on the completed pair sequence	Empirical frequencies $\hat{p}_a$	Laplace-smoothed transition counts $\hat{Q}$

Prediction-space metrics. For linear regression, a length-ℓ evaluation prompt is $c = ((x_1, y_1), \dots, (x_\ell, y_\ell))$. Let $\hat{y}_k(c)$ be T_ϕ's mean prediction of y_k given the prefix $((x_1, y_1), \dots, (x_{k-1}, y_{k-1}), x_k)$, and $\hat{y}_k^{\mathrm{mem}}(c), \hat{y}_k^{\mathrm{gen}}(c)$ the corresponding posterior predictive means under the priors Π_M and Π_∞ respectively. Following Raventós et al. [35], we report the mean-squared gap, averaged over positions, between the transformer's predictions and a baseline's predictions:

$$\Delta_{\mathrm{MSE}} := \mathbb{E}_{c \sim P}\left[\frac{1}{\ell}\sum_{k=1}^{\ell}\left(\hat{y}_k(c) - \hat{y}_k^{\mathrm{ref}}(c)\right)^2\right], \tag{9}$$

where ref refers either to mem or gen. We evaluate this gap on either in-distribution ($P = \mathbb{P}_M$) or out-of-distribution ($P = \mathbb{P}_\infty$) data. We write $\mathbb{P}_\infty$ for the joint law (2) under $\Pi = \Pi_\infty$. The empirical estimator of (9) is given in Appendix F.

For balls-and-urns, let $p_k(c)$ denote the BFT's next-token distribution at position k of prompt $c = (y_1, \ldots, y_\ell)$, and $p_k^{\mathrm{ref}}(c)$ the corresponding next-token distribution under the baseline from (7), with ref standing for mem or gen as in (9). The symmetrized-KL prompt gap of Wurgaft et al. [41] is

$$\Delta_{\mathrm{sKL}} \;:=\; \mathbb{E}_{c\sim P}\left[\frac{1}{\ell}\sum_{k=1}^{\ell}\tfrac{1}{2}\big(\mathrm{KL}(p_k(c)\|p_k^{\mathrm{ref}}(c)) + \mathrm{KL}(p_k^{\mathrm{ref}}(c)\|p_k(c))\big)\right], \tag{10}$$

with P as in (9).

For the Markov chains family we use the same Δ_{sKL}, with the baseline taken as each of the memorizing and generalizing posterior predictives of (7). We note that Park et al. [33] report a sequence-level KL metric of the transformer's empirical transition matrix against the data-generating kernel. As we need a way to compare the transformer's predictions against the memorizing and generalizing baselines, we use the Δ_{sKL} instead.

Latent-space metrics. Let $\widehat{\Pi}(\cdot \mid c)$ be the distribution of the PMC samples conditioned on prompt c, and let $\Pi^{\mathrm{ref}}(\cdot \mid c)$ denote either baseline from (7), with ref standing for mem or gen. Both live on $\mathbb{R}^D$, where D is the dimension in which the latent task is represented: $D = |\mathcal{Y}| = 12$ for balls-and-urns (the probability vector), $D = d = 8$ for linear regression, and $D = |\mathcal{Y}|^2 = 100$ for Markov chains (the flattened transition matrix). Our first metric is the squared *energy distance* [36]. For $U, U' \overset{\mathrm{iid}}{\sim} \widehat{\Pi}(\cdot \mid c)$ and $V, V' \overset{\mathrm{iid}}{\sim} \Pi^{\mathrm{ref}}(\cdot \mid c)$, all four independent, it is

$$\mathcal{E}^2(U,V) \;:=\; 2\,\mathbb{E}\|U-V\| \;-\; \mathbb{E}\|U-U'\| \;-\; \mathbb{E}\|V-V'\|. \tag{11}$$

Our second metric compares the same two distributions through their one-dimensional projections. For the same U and V, the *sliced Wasserstein-1 distance* [5, 24] is

$$\mathrm{SW}_1(U,V) \;:=\; \mathbb{E}_{\vartheta\sim\mathrm{Unif}(\mathbb{S}^{D-1})}\, W_1\big(\vartheta^\top U,\, \vartheta^\top V\big), \tag{12}$$

where W_1 denotes the 1-Wasserstein distance between the laws of the scalar projections $\vartheta^\top U$ and $\vartheta^\top V$. For the posterior, we average each distance over prompts $c \sim P$, with prompts originating either from in-distribution ($P = \mathbb{P}_M$) or out-of-distribution ($P = \mathbb{P}_\infty$) data. For the prior, we set $c = \varnothing$ and take the baseline to be Π_M (memorizing) or Π_∞ (generalizing). There is a single prompt, so no averaging over prompts is involved. In practice each expectation in (11) and (12) is estimated from the finite PMC and baseline samples. The estimators are given below.

We do not use KL divergence for latent-space comparisons. The memorizing baseline is supported on only M latent tasks. KL divergence from the PMC distribution to this baseline is infinite whenever the PMC distribution assigns probability to tasks outside that set. For the continuous generalizing baseline, estimating KL would require an additional estimation step, such as estimating a density or entropy from the PMC samples. Energy distance and sliced Wasserstein-1 can be computed directly from samples without density estimation, and their sample estimates remain finite even when the distributions have different supports. The latent-space comparison is not restricted to these two metrics; other suitable sample-based discrepancies could also be used.

Empirical estimators of the latent-space metrics. We compute the latent-space metrics (11) and (12) with the estimators below. For the prior, the reported quantities are

$$\mathcal{E}^2(U,V) \quad \text{and} \quad \mathrm{SW}_1(U,V), \qquad U \sim \widehat{\Pi}(\cdot \mid \varnothing), \quad V \sim \Pi^* \in \{\Pi_M, \Pi_\infty\}, \tag{13}$$

and for the posterior they are averaged over prompts,

$$\begin{aligned} &\mathbb{E}_{c\sim P}\big[\,\mathcal{E}^2(U,V)\big] \quad \text{and} \quad \mathbb{E}_{c\sim P}[\,\mathrm{SW}_1(U,V)\,], \\ &U \sim \widehat{\Pi}(\cdot \mid c), \quad V \sim \Pi^*(\cdot \mid c) \in \{\Pi^{\mathrm{mem}}(\cdot \mid c),\, \Pi^{\mathrm{gen}}(\cdot \mid c)\}. \end{aligned} \tag{14}$$

We estimate each of these distances from two empirical samples: a PMC sample $U = \{u_i\}_{i=1}^R$ and a baseline sample $V = \{v_j\}_{j=1}^{R'}$. We take $R' = R$, except in the rollout-count study of Appendix E.3, which holds $R' = 1024$ fixed. For the prior, we draw U from the PMC prior $\widehat{\Pi}(\cdot \mid \varnothing)$, and we draw V from either the training prior Π_M or the population prior Π_∞. For the posterior, we draw U from the PMC posterior $\widehat{\Pi}(\cdot \mid c)$, and we draw V from either the memorizing posterior $\Pi^{\mathrm{mem}}(\cdot \mid c)$ or the generalizing posterior $\Pi^{\mathrm{gen}}(\cdot \mid c)$.

We estimate the energy distance (11) with the unbiased U-statistic

$$\widehat{\mathcal{E}}^2(U,V) \;=\; \frac{2}{RR'}\sum_{i,j}\|u_i - v_j\| \;-\; \frac{1}{R(R-1)}\sum_{i\neq i'}\|u_i - u_{i'}\| \;-\; \frac{1}{R'(R'-1)}\sum_{j\neq j'}\|v_j - v_{j'}\|. \quad (15)$$

We report $\widehat{\mathcal{E}}^2$ directly rather than its square root.

For the sliced Wasserstein-1 distance (12), let $\vartheta_1, \ldots, \vartheta_L$ be directions drawn iid from $\mathrm{Unif}(\mathbb{S}^{D-1})$. Let $F_{U,s}$ and $F_{V,s}$ be the empirical distribution functions of $\{\vartheta_s^\top u_i\}_{i=1}^R$ and $\{\vartheta_s^\top v_j\}_{j=1}^{R'}$. The estimator is

$$\widehat{\mathrm{SW}}_1(U,V) \;=\; \frac{1}{L}\sum_{s=1}^{L}\int_0^1 \left|F_{U,s}^{-1}(t) - F_{V,s}^{-1}(t)\right| \mathrm{d}t. \quad (16)$$

When $R' = R$, the integral equals the average of $|u_s^{(k)} - v_s^{(k)}|$ over $k = 1, \ldots, R$, where $u_s^{(k)}$ and $v_s^{(k)}$ are the k-th order statistics of the two projected samples. We take $L = 100$.

Empirical estimators of the prediction-space metrics. In (9) and (10), every quantity inside the prompt expectation is computed exactly: $\hat{y}_k(c)$ is the mean of the transformer's predictive distribution, and the baseline predictions and next-token distributions are computed in closed form (Appendices G, H, and I). Only the prompt expectation $\mathbb{E}_{c\sim P}$ requires estimation. Given evaluation prompts $c_1, \ldots, c_{n_P}$ drawn once from P, the estimator of (9) is

$$\widehat{\Delta}_{\mathrm{MSE}} \;=\; \frac{1}{n_P}\sum_{m=1}^{n_P}\frac{1}{\ell}\sum_{k=1}^{\ell}\left(\hat{y}_k(c_m) - \hat{y}_k^{\mathrm{ref}}(c_m)\right)^2, \quad (17)$$

For linear regression the reported values divide (17) by d, so the plotted gap is per coordinate; this rescaling does not affect which baseline is closer. The estimator $\widehat{\Delta}_{\mathrm{sKL}}$ of (10) replaces the squared difference with the symmetrized KL summand. For linear regression and balls-and-urns the same prompt sets estimate the prompt expectations in the posterior latent-space metrics (14). For Markov chains they differ: the prediction-space metric is read from the training evaluator, which averages the symmetrized KL over the positions of full-length sequences rather than over length-ℓ prompts, so its conditioning length is not ℓ. The number n_P and length ℓ of evaluation prompts for each family are recorded in Appendices G, H, and I.

Reported figures and 1-D slices. For each task family, three sets of figures are reported: PMC prior and posterior samples; task diversity in the prior and posterior; and transient generalization in the posterior. Posterior-space sweeps and dynamics appear in Section 5; prior-space sweeps and samples appear in the family-specific appendices. Because the full $(M, \text{training step})$ grid is large, each of the latter two figures is a 1-D slice. For the *task-diversity sweep*, we sweep all M at the final checkpoint. For the *transient-generalization experiment*, we fix an intermediate task diversity M where transient generalization is most pronounced and sweep all saved training checkpoints. Table 7 records the specific choices for each family.

Table 7: **Experimental slices.** For each task family and phenomenon we report results along a 1-D slice of the task-diversity $\times$ training-step space, sweeping one axis and fixing the other.

Phenomenon (swept axis)	**Task family**	**Fixed axis**
Task-diversity threshold (all M)	Linear regression	final checkpoint
	Balls-and-urns	final checkpoint
	Markov chains	final checkpoint
Transient generalization (all checkpoints)	Linear regression	$M = 32$
	Balls-and-urns	$M = 32$
	Markov chains	$M = 8$

G Linear regression

Data-generating setup. We study in-context linear regression with input dimension $d = 8$ and observation model

$$y_i = w^\top x_i + \varepsilon_i, \qquad x_i \sim \mathcal{N}(0, I_d), \qquad \varepsilon_i \sim \mathcal{N}(0, \sigma^2),$$

with $\sigma^2 = 0.25$. The population prior is $\Pi_\infty = \mathcal{N}(0, I_d)$. At task diversity M, training-sequence generation follows Section 3: sample M regression vectors from Π_∞ to form the training prior, then generate training sequences from that discrete prior. Evaluation prompts are written as

$$c = ((x_1, y_1), \ldots, (x_\ell, y_\ell)),$$

with the latent regression vector drawn either from the training prior Π_M (in-distribution) or from Π_∞ (out-of-distribution).

Baselines. Let the support points of the empirical training prior at diversity M be $w^{(1)}, \ldots, w^{(M)}$. For a prompt $c = ((x_1, y_1), \ldots, (x_\ell, y_\ell))$, write $X = (x_1, \ldots, x_\ell)^\top \in \mathbb{R}^{\ell \times d}$ and $\mathbf{y} = (y_1, \ldots, y_\ell)^\top$. The memorizing baseline uses the empirical training prior, so its posterior is discrete:

$$\Pi^{\mathrm{mem}}(w^{(m)} \mid c) = \frac{\exp\left(-\frac{1}{2\sigma^2}\|\mathbf{y} - Xw^{(m)}\|^2\right)}{\sum_{j=1}^{M} \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{y} - Xw^{(j)}\|^2\right)}.$$

Its Bayes predictor at a query covariate x_* is the prediction under the posterior mean, called dMMSE (discrete minimum mean squared error) by Raventós et al. [35]:

$$\hat{w}_{\mathrm{dMMSE}} = \sum_{m=1}^{M} \Pi^{\mathrm{mem}}(w^{(m)} \mid c)\, w^{(m)}, \qquad \hat{y}_{\mathrm{dMMSE}} = x_*^\top \hat{w}_{\mathrm{dMMSE}}.$$

The Gaussian generalizing baseline uses

$$w \sim \mathcal{N}(0, I_d),$$

which yields the conjugate posterior

$$w \mid X, \mathbf{y} \sim \mathcal{N}(\mu_n, \Sigma_n), \qquad \Sigma_n = (X^\top X/\sigma^2 + I_d)^{-1}, \qquad \mu_n = \Sigma_n X^\top \mathbf{y}/\sigma^2,$$

and the ridge-regression Bayes predictor

$$\hat{w}_{\mathrm{ridge}} = (X^\top X + \sigma^2 I_d)^{-1} X^\top \mathbf{y}.$$

Transformer architecture. The transformer architectures for linear regression in Raventós et al. [35], Carroll et al. [7] and Wurgaft et al. [41] are all decoder-only, sequential residual stream, autoregressive transformers. Wurgaft et al. [41] uses the GPT-NeoX architecture [4] from Huggingface, which implies the use of rotary positional encodings (RoPE) instead of learned absolute positional embeddings. We train transformers without positional encoding. We use the HookedTransformer in TransformerLens [28] as the base implementation for our transformers, and follow the configuration of Carroll et al. [7]: 2 layers with embedding dimension $d_{\mathrm{model}} = 512$, 4 attention heads with dimension $d_{\mathrm{head}} = 128$, MLP hidden dimension $d_{\mathrm{mlp}} = 512$, and GELU activations. This comes to approximately 3.5M parameters per model. We train on sequences of 256 covariate–response pairs. As the covariate and response are modeled separately, this means a context length of 512 tokens.

The transformers in Raventós et al. [35] and Wurgaft et al. [41] utilize a point prediction head, where the transformer outputs a point prediction based on the context, and is trained on autoregressive mean-squared error loss. To perform PMC, we require our transformers to model a predictive distribution. To do so, we follow the setup of Müller et al. [26]: we discretize the output space into categorical buckets and train on (autoregressive) negative log-likelihood. This allows us to sample from the transformer's predictive distribution, and allows for the autoregressive generation of sequences from the model needed for PMC. We discretize y into 256 uniform-width bins covering $[-10, 10]$, with half-normal tails at the two edge bins to have an unbounded support. For prediction space metrics, we obtain a point prediction by taking the expected value of the transformer's predictive distribution. We use inverse transform sampling to sample from the transformer's predictive distribution for PMC rollout.

We also alter the attention masking scheme. We construct the sequence embeddings such that even-position (under 0-indexing) tokens contain the embedding for only the covariates x_i. The odd-position tokens contain the sum of the embeddings of the covariate x_i and the corresponding response y_i. Instead of a traditional causal mask, we instead enforce tokens to attend *only* to themselves and previous *odd-position* tokens in the sequence. This attention masking scheme enforces the relationship that query tokens only attend to themselves and previous complete pairs instead of previous query

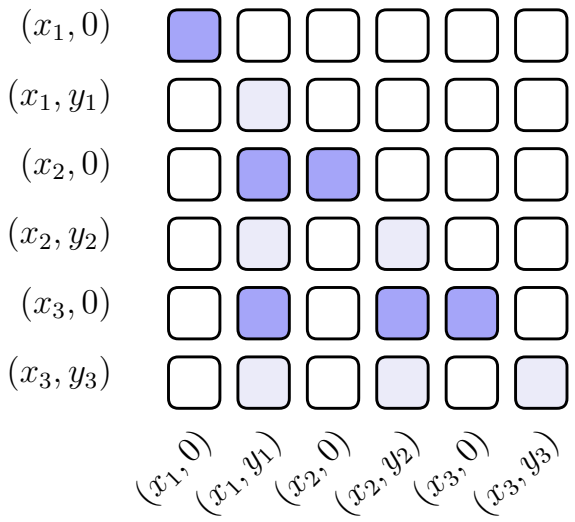


Figure 16: **Autoregressive-PFN attention mask.** Even-position tokens embed the covariate x_i and odd-position tokens embed the pair (x_i, y_i). Each token attends to itself (darker cells) and to all earlier odd-position (full-pair) tokens (lighter cells).

tokens. Without enforcing this structure, the model would have to learn the relationship between token positions and the corresponding covariate-response pair. Figure 16 illustrates the attention mask. We note that the transformers are only trained on outputs from the even-position (query) tokens.

We do not use a beginning-of-sequence (BOS) token in linear regression. The discrete task family transformers (balls-and-urns and Markov chains) use a BOS token for two reasons: it lets the BFT express a marginal predictive $T_\phi(\cdot \mid \varnothing)$ for the first observation, and it supplies the starting context for PMC rollouts from the prior $\widehat{\Pi}(\cdot \mid \varnothing)$. As our sequences are initialized by an initial covariate x_1, this covariate can act in place of the BOS token. To initialize a PMC rollout from the prior, we sample $x_1 \sim \mathcal{N}(0, I_d)$ and feed it in as the first token.

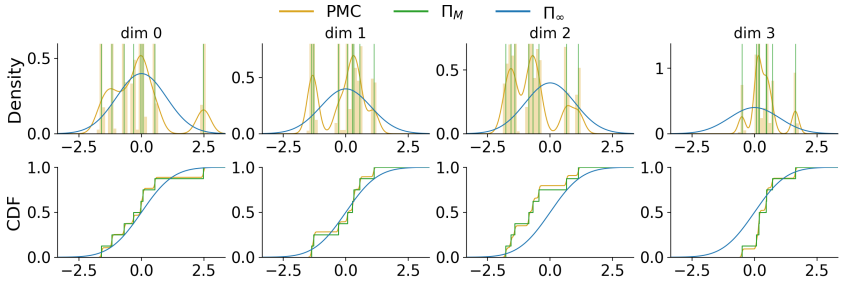
Training details. We optimize with AdamW, warming the learning rate up linearly over $1{,}000$ steps to 10^{-4} and holding it constant thereafter. We train with weight decay 10^{-5}, batch size 256, gradient clipping at norm 1, and $150{,}000$ training steps on a single NVIDIA RTX 5090 GPU with seed 42. Each run takes approximately 90 minutes, and we parallelize the task-diversity sweep across an $8\times$RTX 5090 machine. Each training step draws a fresh batch of regression vectors from the empirical training prior Π_M together with fresh observations conditional on those vectors, and we minimize the cross-entropy of the bin label of each y_i. Checkpoints are saved every 500 steps. The task-diversity sweep reads only the final one, but the transient-generalization experiment sweeps all saved checkpoints at $M = 32$, so that model is trained separately with checkpoints on the union of 200 logarithmically-spaced steps and the same every-500 schedule, which resolves the early steps a linear schedule alone would miss. We train one model per task diversity $M \in \{2^0, 2^1, \ldots, 2^{16}\}$.

PMC. To roll out a sequence from a prompt, we autoregressively sample a fresh covariate $x_i \sim \mathcal{N}(0, I_d)$, append it to the context, query the transformer for its categorical predictive over y_i, and draw y_i from the transformer's predictive distribution through inverse-transform sampling. We append (x_i, y_i) and iterate for N steps. Prior rollouts have no observed prompt and initialize the sequence with a single $x_1 \sim \mathcal{N}(0, I_d)$. Posterior rollouts initialize the sequence with an observed prompt $(x_{1:\ell}, y_{1:\ell})$ drawn from the prompt source under evaluation.

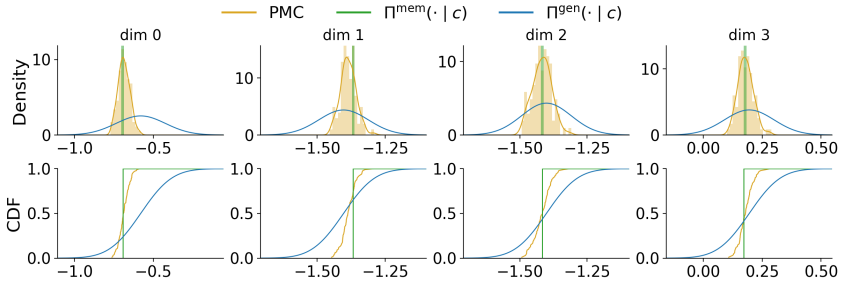
For the task-diversity sweep (Figures 17 and 18) we set the PMC rollout length to $N = 223$ predictive steps. The PMC posterior uses $R = 100$ rollouts per evaluation prompt. We use 128 evaluation prompts at length $\ell = 32$ from each prompt source (Section 5). The PMC prior uses $R = 1024$

rollouts. The transient-generalization experiment at $M = 32$ (Figure 19) uses the same PMC settings and the same evaluation prompts.

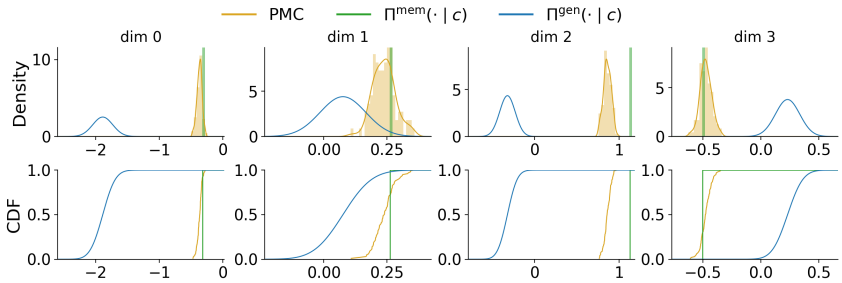
Prior in prediction space. Figure 18(a), leftmost panel, reports the Δ_{MSE} for the transformer's prior. To compute this, we sample 4096 query covariates $x \sim \mathcal{N}(0, I_d)$ and record the unconditioned prediction $\hat{y}(x)$ from the transformer. Δ_{MSE} is the mean squared difference between $\hat{y}(x)$ and the prior predictor of each baseline at the same x. The generalizing predictor is 0, and the memorizing predictor is $\frac{1}{M}\sum_{m=1}^{M}(w^{(m)})^\top x$.



(a) Prior samples.



(b) Posterior samples, conditioned on an in-distribution prompt of length 32.



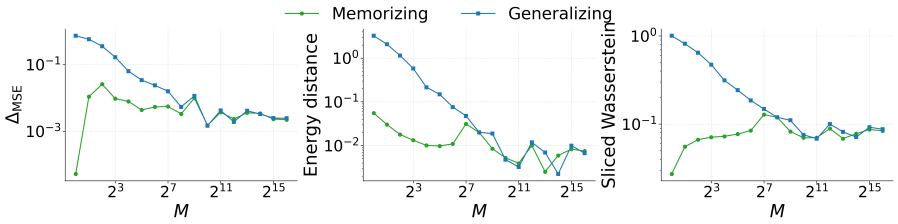
(c) Posterior samples, conditioned on an out-of-distribution prompt of length 32.

Figure 17: **Linear regression: PMC prior and posterior samples.** Marginal densities and CDFs over the first four dimensions of w at task diversity $M = 8$, against the training prior Π_M, the population prior Π_∞, and their corresponding posteriors.

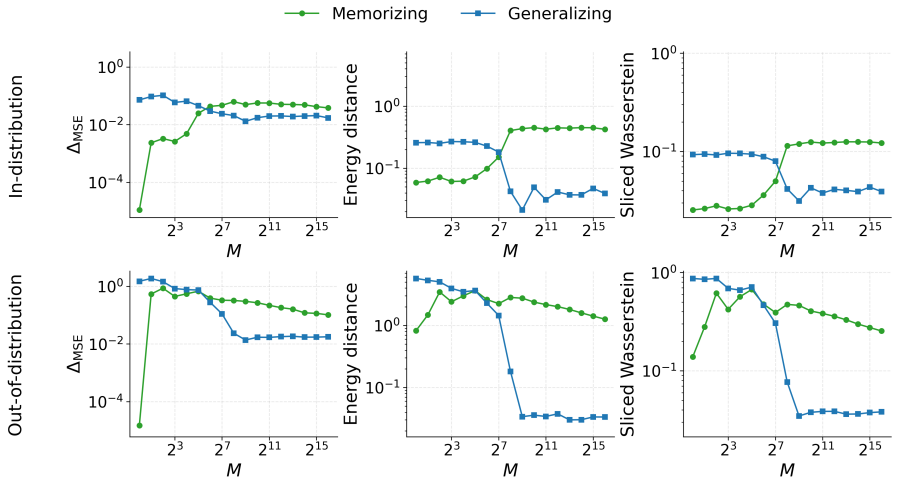
G.1 Distinguishing prior families in latent space

We give a case where the prediction-space comparison cannot separate two priors even in principle, while the latent comparison can.

Prediction space is blind to prior shape. Δ_{MSE} (9) compares point predictions of the response. Given no observations, the prediction under a prior on the coefficients is $\mathbb{E}[\tilde{w}]^\top x$, which depends on the prior only through its mean. A zero-mean Gaussian prior and a zero-mean Laplace prior therefore produce identical predictions at every query x, and Δ_{MSE} between them is exactly zero.



(a) Prior space. Prediction-space Δ_{MSE} (left), energy distance between PMC prior samples and Π_M or Π_∞ (middle), and sliced Wasserstein distance to Π_M or Π_∞ (right), as task diversity varies.



(b) Posterior space. For in-distribution prompts (top) and out-of-distribution prompts (bottom), panels show prediction-space Δ_{MSE} (left), energy distance to the memorizing or generalizing posterior (middle), and sliced Wasserstein distance to the memorizing or generalizing posterior (right), as task diversity varies.

Figure 18: **Linear regression: task diversity in the prior and posterior.** (a) At small task diversity, the PMC prior is much closer to the training prior Π_M than to Π_∞. As M grows, the two distances converge and coincide by $M = 2^{10}$, as Π_M approaches Π_∞, so the prior shows no distinct switch. (b) The PMC posterior switches from the memorizing to the generalizing baseline. For in-distribution prompts, energy distance and sliced Wasserstein switch between $M = 2^7$ and 2^8, and Δ_{MSE} between 2^5 and 2^6. For out-of-distribution prompts, the two baselines are about equally distant at intermediate M, and the generalizing baseline is clearly closer from $M = 2^8$. Figure 4(a) shows the Δ_{MSE} and energy distance columns.

Setup. We compare the PMC prior against two priors on $\mathbb{R}^8$: the population prior $\Pi_\infty = \mathcal{N}(0, I)$, and a Laplace prior with independent coordinates and scale $b = 1/\sqrt{2}$. Since $\mathrm{Var}[\mathrm{Laplace}(0, b)] = 2b^2$, the two priors share the same mean and variance and differ only in shape. PMC prior samples come from $R = 8192$ rollouts of length $N = 223$; $\hat{w}$ is the least-squares estimate on each rollout. We draw 8192 samples from each prior. Energy distance is evaluated on 4096 samples because it is a U-statistic over all pairs; sliced Wasserstein-1 uses the full set. We report the *margin*, the distance to the Laplace prior minus the distance to the Gaussian prior, so a positive margin means the Gaussian is closer. Standard errors come from 32 disjoint blocks, each scored against its own samples of the two priors.

Calibration. The two priors are close together, so the comparison needs a scale. We repeat the measurement on a true Gaussian sample of the same size, which gives a calibration value for the margin at this sample budget. That oracle separates the two priors by 7.8 to 13.9 standard errors in sliced Wasserstein. The calibration is not optional: at $R = 1024$, the number of rollouts used elsewhere in this paper, the sliced Wasserstein oracle falls to about two standard errors, which would leave this comparison without the resolution to distinguish the two priors. We quote standard errors for sliced Wasserstein only. Energy distance is a U-statistic whose variance becomes degenerate as

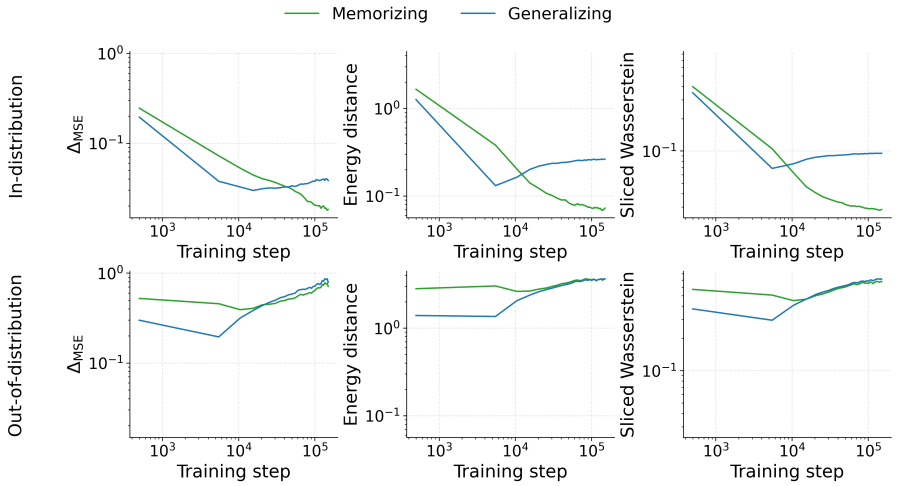


Figure 19: **Linear regression: transient generalization.** At intermediate task diversity $M = 32$, for in-distribution prompts (top), the transformer first approximates the generalizing baseline and then specializes to the memorizing baseline, from about 10^4 training steps in energy distance and sliced Wasserstein and later in Δ_{MSE}. For out-of-distribution prompts (bottom), the early match to the generalizing baseline is lost by about 2×10^4 steps; the two distances then grow together and nearly coincide, so neither baseline is matched. Columns show prediction-space Δ_{MSE} (left), energy distance to each baseline (middle), and sliced Wasserstein distance to each baseline (right). Figure 4(b) shows the Δ_{MSE} and energy distance columns.

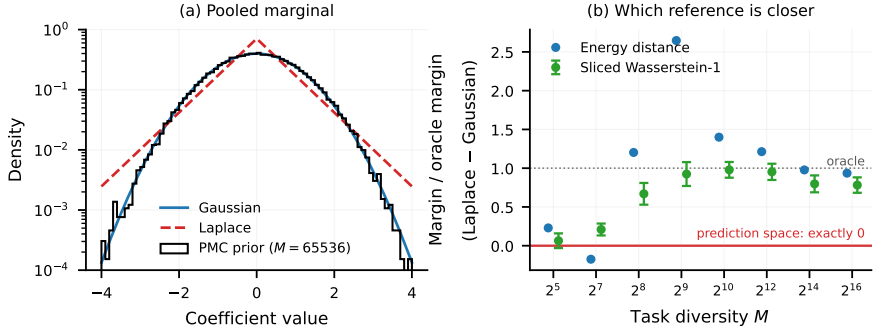


Figure 20: **PMC separates prior families that the prediction-space comparison cannot.** Linear regression on $\mathbb{R}^8$, PMC prior from $R = 8192$ rollouts of length $N = 223$, compared against $\Pi_\infty = \mathcal{N}(0, I)$ and a variance-matched $\mathrm{Laplace}(0, 1/\sqrt{2})$. **(a)** Coordinates pooled at $M = 65536$ on a log density axis: the PMC prior follows the Gaussian into the tails and departs from the Laplace, which the shared mean and variance leave as the only distinguishing feature. **(b)** Margin, the distance to the Laplace prior minus the distance to the Gaussian, divided by the oracle margin that a true Gaussian sample of the same size achieves; positive means the Gaussian is closer and 1 is the oracle. Samples are rescaled to unit variance so the comparison measures shape rather than scale. Error bars are block standard errors and are shown for sliced Wasserstein-1 only; the energy distance is a U-statistic whose variance degenerates as the samples approach the prior, so its margins are point estimates. Δ_{MSE} between the two priors is exactly zero at every task diversity and every rollout budget, since both have mean zero: the red line is the entire prediction-space comparison.

the two samples approach the same law, so the block estimate above does not extrapolate from block size to the full sample, and we report its margins as point estimates.

Scale must be controlled. The PMC prior is under-dispersed at every task diversity: $\mathrm{Var}[\hat{w}]$ is 0.919 at $M = 32$, falls to 0.746 at $M = 256$, and lies between 0.80 and 0.88 for $M \geq 512$. Both priors have unit variance, so a shrunk sample is penalized for scale as well as shape, and a distribution with less spread places more mass near zero, as the Laplace density does relative to the Gaussian. Compared without this correction the PMC prior appears closer to the Laplace prior at every task diversity, by as much as 10.4 standard errors. We therefore rescale the PMC samples to unit variance before comparing, which isolates shape.

Result. After rescaling, the Gaussian prior is closer from $M = 256$ onward, in both metrics and at every larger task diversity tested: sliced Wasserstein gives margins of 4.8 to 9.6 standard errors over $M = 256, \ldots, 65536$, and the energy-distance margins are positive throughout that range. At $M = 32$ neither metric separates the two priors, and at $M = 128$ only sliced Wasserstein does (2.7 standard errors). This is expected, since the training prior Π_M is supported on 32 or 128 atoms and is close to neither continuous prior. From $M = 512$ the sliced Wasserstein margin reaches 78 to 98 percent of the oracle, so the PMC prior separates the two priors almost as well as a true Gaussian sample of the same size. After rescaling, PMC therefore distinguishes prior families that the prediction-space comparison cannot distinguish at all. The unrescaled samples do not support this conclusion, as the previous paragraph records. Figure 20 shows both halves of the comparison.

H Balls-and-urns

Data-generating setup. The balls-and-urns family generalizes the Beta-Bernoulli setup of Appendix B from two outcomes to $|\mathcal{Y}| = 12$ categorical outcomes. Let $p \in \Delta^{|\mathcal{Y}|-1}$ denote the urn proportions, with population prior $\Pi_\infty = \mathrm{Dirichlet}(1, \ldots, 1)$. At task diversity M, training-sequence generation again follows Section 3: sample M urns from Π_∞ to form the empirical training prior, then generate training sequences from that discrete prior. Evaluation prompts are written as $c = (y_1, \ldots, y_\ell)$, with the latent urn drawn either from the empirical training prior Π_M (in-distribution) or from Π_∞ (out-of-distribution).

Baselines. Let the support points of the empirical training prior at diversity M be $p^{(1)}, \ldots, p^{(M)}$. For a prompt $c = (y_1, \ldots, y_\ell)$, the memorizing baseline uses that empirical training prior, so the posterior is the discrete law

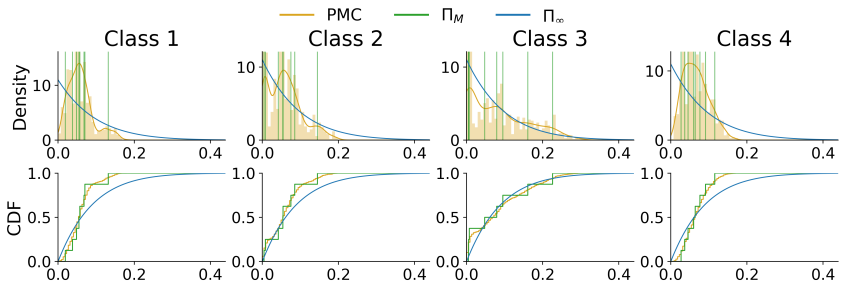
$$\Pi^{\mathrm{mem}}(p^{(m)} \mid c) = \frac{\prod_{i=1}^{\ell} p_{y_i}^{(m)}}{\sum_{j=1}^{M} \prod_{i=1}^{\ell} p_{y_i}^{(j)}}.$$

The generalizing baseline uses the Dirichlet population prior, giving the conjugate posterior $p \mid c \sim \mathrm{Dirichlet}(1 + N_1(c), \ldots, 1 + N_{|\mathcal{Y}|}(c))$, where $N_a(c) = \sum_{i=1}^{\ell} \mathbf{1}\{y_i = a\}$.

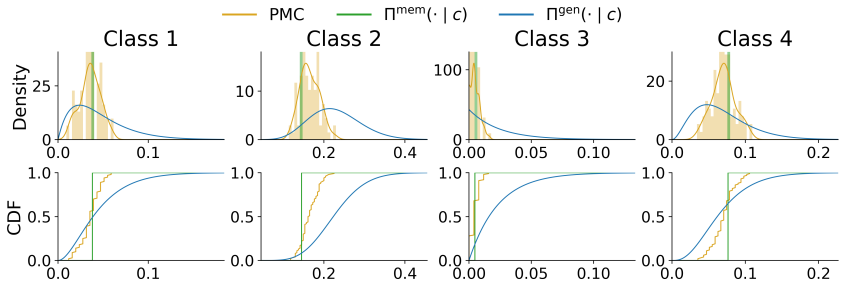
Transformer architecture. We follow the architecture of Wurgaft et al. [41], who train decoder-only autoregressive transformers on sequences of categorical observations. We use the HookedTransformer in TransformerLens [28] as the base implementation for our transformers, and train without positional encoding. We use 2 layers with embedding dimension $d_{\mathrm{model}} = 64$, a single attention head with dimension $d_{\mathrm{head}} = 64$, MLP hidden dimension $d_{\mathrm{mlp}} = 256$, and GELU activations. This comes to approximately 118K parameters per model. We train with maximum context length 256, consisting of the BOS token followed by 255 categorical observations.

We use the standard left-to-right causal attention mask and train on autoregressive cross-entropy loss. To model the unconditioned prior predictive distribution, we prepend a beginning-of-sequence (BOS) token to each sequence, which lets the BFT express a marginal predictive $T_\phi(\cdot \mid \varnothing)$ for the first observation and supplies the starting context for PMC rollouts from the prior $\widehat{\Pi}(\cdot \mid \varnothing)$. We do not include the BOS token in the output vocabulary space. That is, we have the input vocabulary dimension $d_{\mathrm{vocab_in}} = 13$ and output vocabulary $d_{\mathrm{vocab_out}} = 12$.

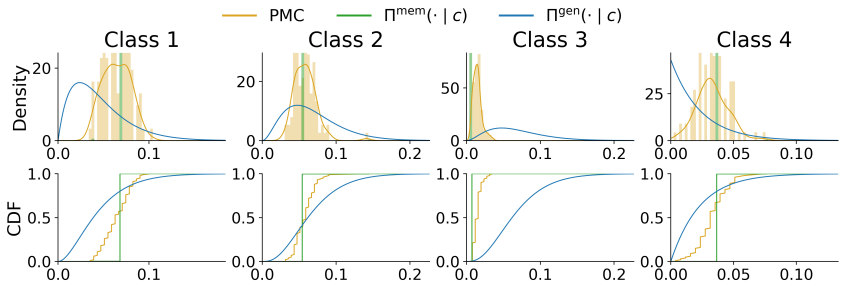
Training details. We optimize with AdamW at learning rate 5×10^{-4}, weight decay 10^{-5}, batch size 64, gradient clipping at norm 1, and $100{,}000$ training steps on a single NVIDIA RTX 5090 GPU with seed 42. Each run takes approximately 85 minutes, and we parallelize the task-diversity sweep across an $8\times$RTX 5090 machine. We use no warmup or learning-rate decay. Each training step



(a) Prior samples.



(b) Posterior samples, conditioned on an in-distribution prompt of length 32.

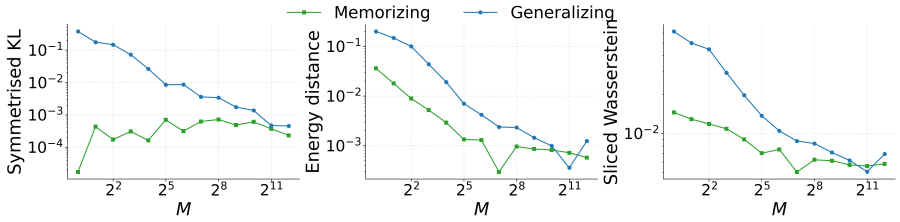


(c) Posterior samples, conditioned on an out-of-distribution prompt of length 32.

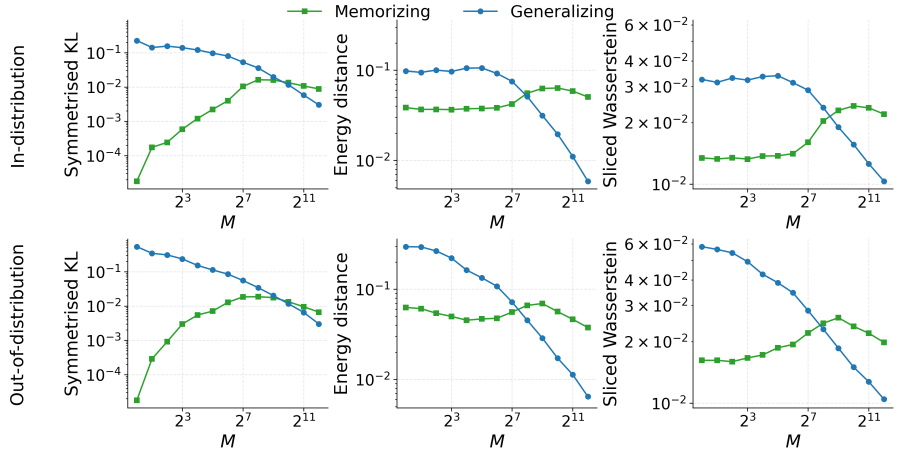
Figure 21: **Balls-and-urns: PMC prior and posterior samples.** Marginal densities and CDFs over the first four classes of p at task diversity $M = 8$, against the training prior Π_M, the population prior Π_∞, and their corresponding posteriors.

draws a fresh batch of urn proportions from the empirical training prior Π_M together with fresh balls conditional on those urns, and we minimize the cross-entropy of the categorical prediction at each position. Checkpoints are saved every 500 steps. The task-diversity sweep reads only the final one, but the transient-generalization experiment sweeps all saved checkpoints at $M = 32$, so that model is trained separately with checkpoints on the union of 200 logarithmically-spaced steps and the same every-500 schedule, which resolves the early steps a linear schedule alone would miss. We train one model per task diversity $M \in \{2^0, 2^1, \ldots, 2^{12}\}$.

PMC. For the task-diversity sweep (Figures 21 and 22) we roll out $N = 255 - \ell$ predictive steps, so every completed sequence contains 255 observations. The PMC posterior uses $R = 100$ rollouts per evaluation prompt. We use 128 evaluation prompts at length $\ell = 32$ from each prompt source. In-distribution prompts are obtained by sampling an urn p from the empirical training prior Π_M and then drawing a length-32 sequence $y_{1:32} \overset{\text{iid}}{\sim} \text{Categorical}(p)$. Out-of-distribution prompts sample p from the population prior $\Pi_\infty = \text{Dirichlet}(1, \ldots, 1)$ instead. The PMC prior uses $R = 1024$ rollouts of length N from the BOS-only prompt. After each rollout we apply the empirical-frequency estimator $\hat{p}_a = (\ell + N)^{-1} \sum_{i=1}^{\ell+N} \mathbf{1}\{y_i = a\}$ to obtain a PMC sample of $\hat{p}$. The transient-generalization experiment at $M = 32$ (Figure 23) uses the same PMC settings and the same evaluation prompts, and evaluates every fifth saved checkpoint from the training run to reduce analysis cost.



(a) Prior space. Prediction-space symmetrized KL gap (left), energy distance between PMC prior samples and Π_M or Π_∞ (middle), and sliced Wasserstein distance to Π_M or Π_∞ (right), as task diversity varies.



(b) Posterior space. For in-distribution prompts (top) and out-of-distribution prompts (bottom), panels show prediction-space symmetrized KL gap (left), energy distance to the memorizing or generalizing posterior (middle), and sliced Wasserstein distance to the memorizing or generalizing posterior (right), as task diversity varies.

Figure 22: **Balls-and-urns: task diversity in the prior and posterior.** Distances from the PMC prior and posterior to the memorizing and generalizing baselines as task diversity varies. We reproduce the task-diversity phenomenon in the prior and posterior, as task diversity increases, PMC samples have smaller distance to the population prior Π_∞ and the generalizing posterior.

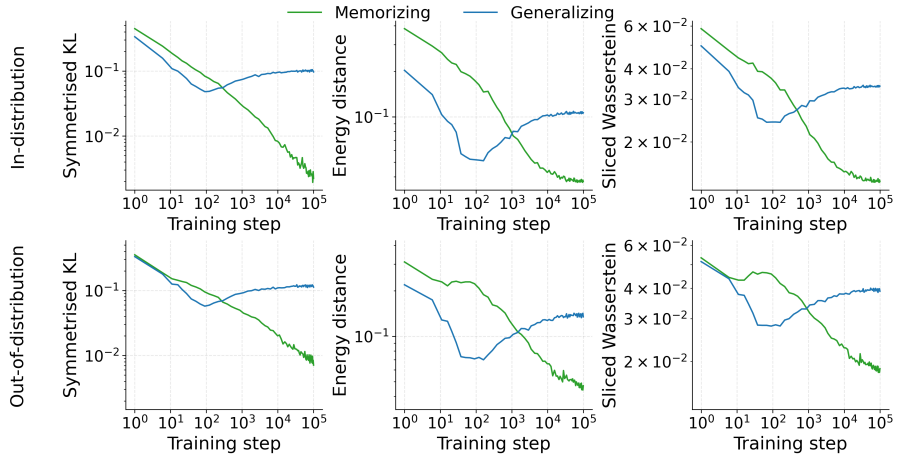


Figure 23: **Balls-and-urns: transient generalization in the posterior.** At intermediate task diversity $M = 32$, the transformer's PMC posterior first approximates the generalizing baseline before later specializing to the memorizing baseline.

Figure 22 shows the task-diversity transition in both prediction space and latent space. The posterior switches in energy distance between $M = 2^7$ and 2^8, and in in-distribution sliced Wasserstein between 2^8 and 2^9, both before the prediction-space metric, which switches between 2^9 and 2^{10}.

Prior in prediction space. Figure 22(a), leftmost panel, reports the symmetrized KL divergence for the transformer's prior. To compute this, we record the unconditioned next-token predictive $\hat{p}$ from the transformer on the BOS token alone. The symmetrized KL divergence is between $\hat{p}$ and the prior predictive of each baseline over the first observation. The generalizing predictive is uniform on $\mathcal{Y}$, and the memorizing predictive is $\frac{1}{M}\sum_{m=1}^{M} p^{(m)}$.

I Markov chains

Data-generating setup. Let Q be a transition matrix on the state space $\mathcal{Y}$. The population prior Π_∞ draws each row of Q independently from $\mathrm{Dirichlet}(1,\dots,1)$. At task diversity M, training-sequence generation follows Section 3: sample M kernels from Π_∞ to form the empirical training prior, then generate training sequences from that discrete prior. Each sequence drawn from Q is initialized with y_1 from the stationary distribution of Q. For the closed-form baselines we drop the $P(y_1 \mid Q)$ factor from the likelihood, leaving the $\ell - 1$ transitions $y_{i-1} \to y_i$ as the contributing terms. This ensures the posteriors are conjugate, and matches the convention of Section 2.2, where Y_1 is taken as given. The omission is not innocuous at our prompt length: the sequences are generated with y_1 drawn from the stationary distribution of Q, so the exact posterior under the generative process carries a stationary-probability factor that the baselines omit, one of $\ell = 32$ likelihood contributions, the initial state and 31 transitions. This is one candidate contribution to the looser match reported below. Evaluation prompts are written as $c = (y_1, \dots, y_\ell)$, with the latent kernel drawn either from the empirical training prior Π_M (in-distribution) or from Π_∞ (out-of-distribution).

Baselines. The generalizing baseline uses Π_∞ as the prior. Its posterior is

$$\Pi^{\mathrm{gen}}(Q \mid c) \propto \prod_{i=2}^{\ell} Q_{y_{i-1},y_i}\, \Pi_\infty(Q).$$

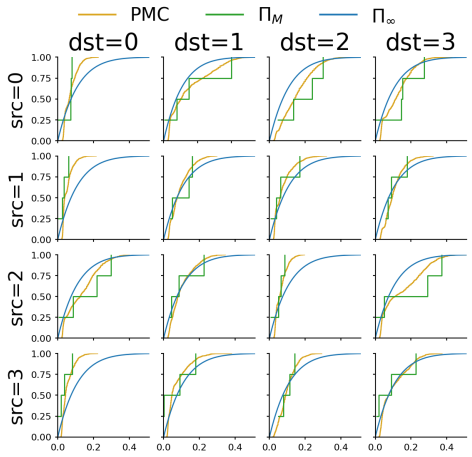
Let the support points of the empirical training prior at diversity M be $Q^{(1)}, \dots, Q^{(M)}$. For a prompt $c = (y_1, \dots, y_\ell)$, the memorizing baseline uses the empirical training prior. The posterior is the discrete law

$$\Pi^{\mathrm{mem}}(Q^{(m)} \mid c) = \frac{\prod_{i=2}^{\ell} Q^{(m)}_{y_{i-1},y_i}}{\sum_{j=1}^{M} \prod_{i=2}^{\ell} Q^{(j)}_{y_{i-1},y_i}}.$$

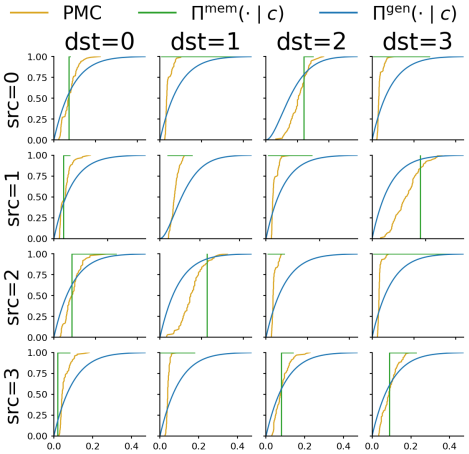
The construction mirrors the memorizing and generalizing baselines used for balls-and-urns and linear regression.

Transformer architecture. For the Markov chains family, we use the same architecture as in Park et al. [33], namely a decoder-only causal transformer. We implement the model directly in PyTorch. We use 2 pre-norm decoder blocks, multi-head causal self-attention with 4 heads, model hidden dimension $d_{\mathrm{model}} = 64$, MLP hidden dimension $d_{\mathrm{mlp}} = 256$, and ReLU activations. Positional information is encoded with rotary embeddings (RoPE) at base 10^4. Each training sequence consists of 256 sampled states; the model input is the BOS token followed by the first 255 states (an input window of 256 tokens), with all 256 states as prediction targets. Unlike the linear regression models, which predict a continuous output distribution, the Markov chains model operates on discrete state tokens with $|\mathcal{Y}| = 10$ states. The input vocabulary has size $|\mathcal{Y}| + 1 = 11$ to accommodate a BOS token, and the linear output head produces a matching 11 logits. At inference we discard the BOS logit so the model only predicts states in $\mathcal{Y}$. Training cross-entropy is computed over all 11 logits, with targets always in $\mathcal{Y}$.

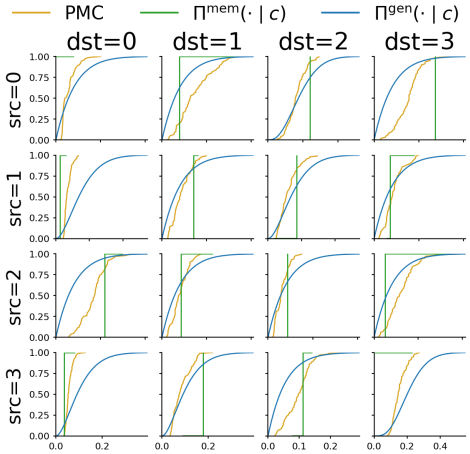
Training details. We optimize with AdamW at constant learning rate 6×10^{-4}, weight decay 0.01, batch size 128, and 100,000 training steps on a single NVIDIA RTX 5090 GPU with seed 42. Each run takes approximately 150 minutes, and we parallelize the task-diversity sweep across an 8×RTX 5090 machine. We use no warmup, no learning-rate decay, and no gradient clipping. Each training step draws a fresh batch of transition matrices from the empirical training prior Π_M together with fresh sequences conditional on those kernels, and we minimize the cross-entropy of the categorical prediction at each position. We save checkpoints every 200 training steps. We train one



(a) Prior samples.



(b) Posterior samples, conditioned on an in-distribution prompt of length 32.

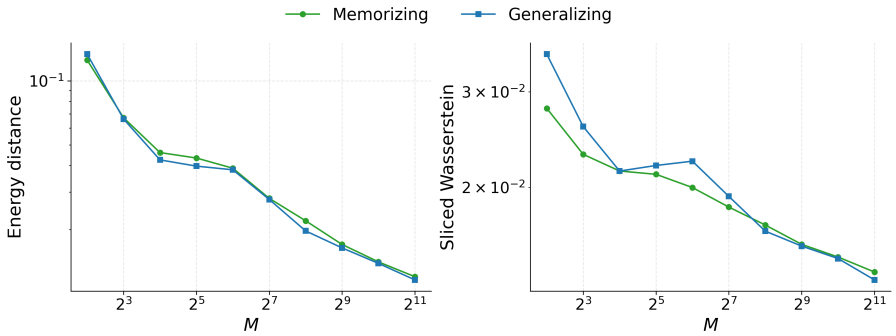


(c) Posterior samples, conditioned on an out-of-distribution prompt of length 32.

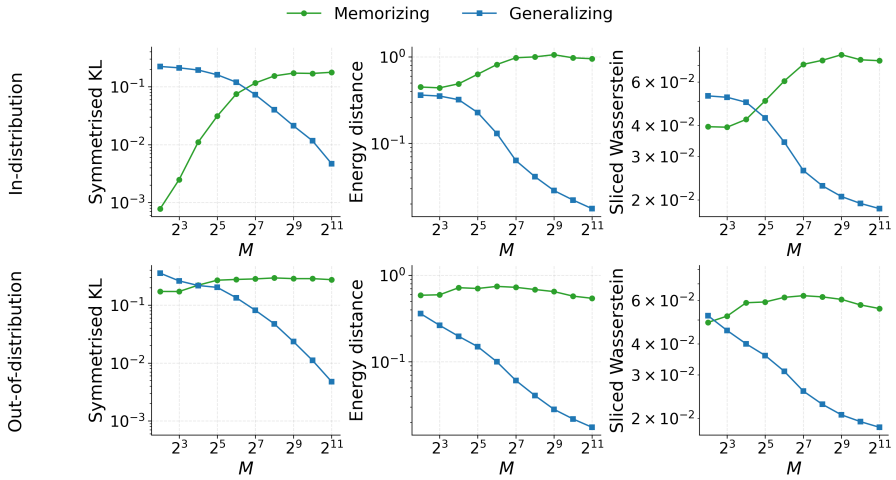
Figure 24: **Markov chains: PMC prior and posterior samples.** Each figure is a 4×4 matrix of panels mirroring the top-left block of the transition matrix: the panel in row $\mathrm{src} = a$ and column $\mathrm{dst} = b$ shows the marginal CDF of the transition probability $Q_{a,b}$ from state a to state b, at task diversity $M = 4$, against the training prior Π_M, the population prior Π_∞, and their corresponding posteriors.

model per task diversity $M \in \{2^2, 2^3, \ldots, 2^{11}\}$. Two evaluations run during training. The first, a training monitor in the style of Park et al. [33] that does not enter any reported figure, computes a stationary-weighted KL between the transformer's implied transition matrix and the data-generating kernel every 200 steps: each row $\hat{Q}_{a,\cdot}$ is read off the model's next-token predictive after a length-200 context ending in state a, and the divergence is averaged over 10 evaluation chains drawn either from the training pool (in-distribution) or afresh from Π_∞ (out-of-distribution). The second computes the prediction-space Δ_{sKL} of Appendix F against the memorizing and generalizing baselines on 16 full-length sequences per evaluation point; the batch is kept small because the memorizing baseline's likelihood computation scales with M.

PMC. For the task-diversity sweep and the transient-generalization experiment, we set the PMC rollout length to $N = 223$ predictive steps. The PMC posterior uses $R = 100$ rollouts per evaluation prompt. We use 128 evaluation prompts at length $\ell = 32$ from each prompt source. In-distribution prompts are obtained by sampling a transition matrix from the training pool and generating a 32-state sequence from it. Out-of-distribution prompts are obtained by sampling a fresh transition matrix



(a) Prior space. Energy distance between PMC prior samples and Π_M or Π_∞ (left), and sliced Wasserstein distance to Π_M or Π_∞ (right), as task diversity varies.



(b) Posterior space. For in-distribution prompts (top) and out-of-distribution prompts (bottom), panels show prediction-space symmetrized KL gap (left), energy distance to the memorizing or generalizing posterior (middle), and sliced Wasserstein distance to the memorizing or generalizing posterior (right), as task diversity varies.

Figure 25: **Markov chains: task diversity in the prior and posterior.** Distances from the PMC prior and posterior to the memorizing and generalizing baselines as task diversity varies.

whose rows are drawn independently from $\mathrm{Dirichlet}(1, \ldots, 1)$ and then generating a sequence of length 32 from the drawn matrix. The PMC prior uses $R = 1024$ rollouts with the BOS-only prompt and rollout length N. After each rollout, we Laplace-smooth the empirical transition counts of the full state sequence. For prior PMC this is the N generated tokens (excluding the BOS token). For posterior PMC it is the ℓ prompt tokens followed by the N generated tokens. For each pair $(a, b) \in \mathcal{Y} \times \mathcal{Y}$,

$$\hat{Q}_{a,b} = \frac{1 + N_{a,b}}{|\mathcal{Y}| + N_{a,\cdot}},$$

where $N_{a,b}$ counts $a \to b$ transitions across this full sequence and $N_{a,\cdot} = \sum_{b' \in \mathcal{Y}} N_{a,b'}$ counts visits to state a.

PMC samples track the baselines less tightly in the Markov chains family than in linear regression or balls-and-urns (Figure 24). We propose two reasons for this. First, the transition matrix Q has $|\mathcal{Y}|(|\mathcal{Y}| - 1) = 90$ free parameters, considerably more than the latents in the other two families, so the PMC posterior is higher-dimensional. Second, relative to its latent task the Markov chains BFT is the most heavily loaded model in our study: it must track a 90-parameter latent with roughly the same parameter budget ($\approx 10^5$) that the balls-and-urns BFT devotes to an 11-parameter latent.

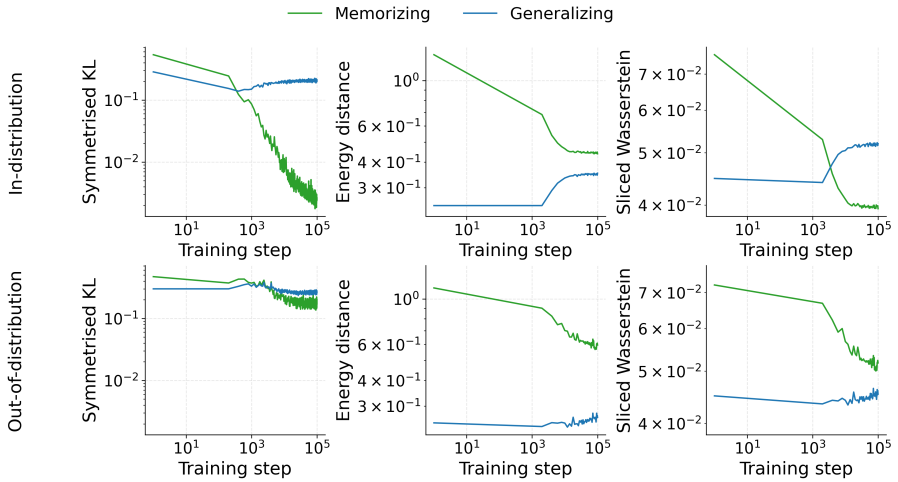


Figure 26: **Markov chains: transient generalization in the posterior.** At task diversity $M = 8$ and for in-distribution prompts, the transformer first approximates the generalizing baseline before later specializing to the memorizing baseline, in prediction space (symmetrized KL) and in the PMC posterior under sliced Wasserstein. Under energy distance, the generalizing baseline stays closer throughout training.

J Marginal densities and CDFs from PMC

The marginal densities and CDFs of the PMC samples for all three task families and every task diversity are at https://bft-pmc.github.io/. Representative examples appear in the main text and in the family appendices above.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the goal of using predictive Monte Carlo (PMC) to interpret Bayes-filtered transformers by recovering their implicit prior and posterior beliefs in latent space. This goal is well aligned with the empirical claims, which evaluate PMC on three Bayes-filtered transformer task families: linear regression, balls-and-urns, and Markov chains.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper includes a Limitations paragraph in the conclusion explaining that the applicability of PMC to real language models remains unclear, because the sufficient conditions for validity may be too strong for frontier LLMs.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not introduce new theoretical results. It relies on established theorems and existing literature.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The experimental setup, model architectures, baselines, evaluation metrics, and PMC-related parameters are documented in Section 5 and Appendices F, B, G, H, I.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility.

In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The paper provides a link to a code repository at https://github.com/afiq-aswadi/bft-pmc, containing the implementation needed to reproduce the experiments. The repository includes code for training the Bayes-filtered transformers, running predictive Monte Carlo rollouts, computing the latent-space distance metrics and generating the main experimental figures. The repository also includes the precomputed data needed to redraw the key figures.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies the synthetic task families, model architectures, and training hyperparameters in sufficient detail for reproducibility. Table 5 summarizes the main experimental settings, and additional task family specific details are provided in Appendices F, G, H and I.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

Justification: The paper does not report error bars because the experiments are not primarily aimed at establishing statistically significant performance differences between competing methods or comparing precise individual point estimates. Instead, the main empirical claims focus on qualitative, trend-level phenomena, specifically the task-diversity threshold and transient generalization observed across task-diversity scales and training checkpoints.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendices G, H, I provide the compute resources details, specifying the use of a single NVIDIA RTX 5090 GPU and an 8x RTX 5090 machine for task sweeps along with approximate runtimes.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses synthetic task families and trained transformer models in controlled experiments, and does not involve human subjects, private data, scraped datasets, or high-risk deployment.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Yes. The Conclusion discusses broader societal and safety relevance by proposing that BFTs with latent behavioral states could provide a controlled testbed for studying emergent misalignment and adversarial vulnerabilities, offering a possible framework for analyzing and eventually mitigating related risks in LLMs.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The models presented are restricted to synthetic, low-dimensional statistical tasks and do not pose a high risk for misuse.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper properly credits the tools and prior implementations they built upon, such as the Hooked Transformer in TransformerLens and the GPT-NeoX architectures.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new reusable assets like datasets or codebase libraries.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing, human participants, participant instructions, screenshots, or participant compensation. All experiments use synthetic data-generating processes.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve human subjects research or crowdsourcing, so IRB approval or equivalent review is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development and experiments do not use an LLM as an important, original, or non-standard component.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.