ELICITATION WITHOUT BACKPROP: STEERING MODEL BEHAVIOR BY OPTIMIZING THE LATENT POSTERIOR

Anonymous authors
Paper under double-blind review

ABSTRACT

In the *latent posterior model* of transformer behavior, the next-token distribution arises from a posterior over latent predictive models conditioned on the context, mixed to generate continuations. We exploit this model in settings where it is exact, namely Bayes-filtered transformers (BFTs) meta-trained on sequences from a hierarchical prior, to introduce **Posterior Prefix Tuning (PPT)**, a new method for *eliciting* behavior from a transformer: given a utility function on continuations, find a prompt under which the model's expected utility is large. For a BFT, the elicitation objective factors through the latent posterior, and we show that the gradient of a tilted surrogate is a covariance over the tilted posterior between the per-model utility and the score of the tilt. PPT optimizes the parameters of a distribution over hard prompts, drawing latent-model samples once from the BFT's prior via predictive Monte Carlo (PMC) and computing its gradient by importance sampling. The inner loop performs no transformer forwards and no backpropagation through the model, and the prior samples are utility-independent, so the cost of characterizing the BFT amortizes across utilities. We validate PPT and a Rao–Blackwellized variant on Beta–Bernoulli and reinforced urn BFTs across three utility families (reverse cross-entropy, frequency matching, Dyck validity).

1 INTRODUCTION

Transformers trained on distributions with diverse tasks exhibit *in-context learning* (ICL), the ability to learn from context alone (Brown et al., 2020). One important model of this behavior, the **latent posterior model**, holds that the transformer maintains a posterior over latent predictive models, updated by the tokens it has seen, and uses this posterior to generate continuations (Xie et al., 2022; Panwar et al., 2024; Marks et al., 2026). The latent posterior model has been verified as a good approximation in synthetic settings such as in-context linear regression (Garg et al., 2022; Akyürek et al., 2023), and has the potential to confer new ideas for understanding and controlling transformer behavior more broadly.

In this paper we explore this potential, motivated by the elicitation problem. We currently lack a deep scientific understanding of why transformers, post-trained to be effective assistants, behave (and misbehave) in the ways that they do. These systems undergo testing before deployment, which can be formalized as the evaluation of utility functions U on the model's outputs across a distribution of prompts. However, the space of possible prompts is vast, and it is impossible to establish by exhaustive search that no untested prompt will produce undesirable behavior. This leads to the problem of elicitation (Irving et al., 2025): given a specification of target behavior (formalized as a utility function U), can we find a prompt under which the model's expected utility is large? The utility specifies whichever behavior we seek to elicit; in safety testing, we deliberately target behavior undesirable in deployment so that it can be identified before deployment. This is closely related to the problem of prompt and prefix tuning (Li & Liang, 2021; Lester et al., 2021).

A promising avenue for investigating the elicitation problem is to factor it through the latent posterior. Rather than searching directly over the combinatorial space of prompts, we ask: is there a prompt conditioned on which the latent posterior concentrates on predictive models with a high probability of generating the target behavior? We study this question in **Bayes-filtered transformers** (BFTs): transformers meta-trained on sequences from a two-stage hierarchical process, in which a latent task is first sampled from a prior and the sequence is then generated conditional on the task. For

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

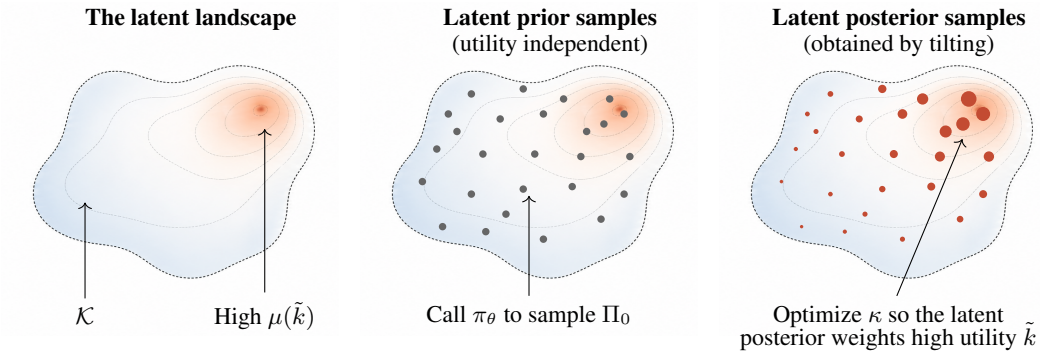


Figure 1: **Latent posterior sampling and tilt for elicitation.** To elicit behavior measured by utility U from the transformer π_θ with latent kernel space $\mathcal{K}$ we look for prompts $z_{1:m}$ which bias the latent posterior distribution $\Pi(d\tilde{k} \mid z_{1:m})$ over $\mathcal{K}$ towards kernels $\tilde{k}$ with high utility $\mu(\tilde{k})$. *Left*: a depiction of the space of latent kernels $\mathcal{K}$. *Middle*: we use Predictive Monte Carlo (PMC) to sample once from the latent prior distribution Π_0. This is the only time we call the transformer π_θ and this doesn't use U. *Right*: we use a continuous surrogate κ for $z_{1:m}$ and a tilt $W(\tilde{k}, \kappa)$ to define a tilted latent posterior $\Pi_{\text{tilt}}(d\tilde{k};\, \kappa)$ samples from which are obtained by importance sampling from our fixed prior samples. We then optimize κ to maximize the posterior expectation of the utility $\mu(\tilde{k}; \kappa)$, which we denote $J_{\text{tilt}}(\kappa)$. Good κ lead to posteriors with high probability on high utility latents. Snapping yields high utility hard prompts $z_{1:m}$.

such sequences, the autoregressive log-loss is minimized at every position by the Bayesian posterior predictive under that prior (Ortega et al., 2019). We call a BFT that attains this minimum exactly the *idealized BFT*; it realizes the latent posterior model by construction, and a suitably trained BFT approximately admits the same factorization. For a BFT, elicitation can therefore be carried out entirely in latent-posterior space. We call the resulting method **Posterior Prefix Tuning** (PPT).

Figure 1 sketches PPT. The method has two properties:

- **No backpropagation through the transformer.** PPT computes its gradient entirely in latent-posterior space, so the inner optimization loop never backpropagates through the transformer and makes zero transformer calls per step.
- **Utility amortization.** The bulk of PPT's computation is utility-independent: characterizing the BFT's beliefs (the prior samples in the middle panel of Figure 1) is a one-time cost, after which any number of utility functions can be optimized against at negligible marginal cost.

We instantiate this approach in two regimes whose latent predictive model admits a finite-dimensional parameterization: exchangeable processes (latent pmf) and 1-Markov exchangeable processes (latent transition matrix). For each, we meta-train a corresponding BFT. We optimize over a kernel from which prompts are drawn (a pmf in the exchangeable case, a transition matrix in the 1-Markov case) and recover a hard prompt from the optimum.

We find that PPT and its Rao–Blackwellized variant PPT-RB are effective across both BFTs, three utility families (reverse cross-entropy, frequency matching, Dyck validity), and prompt lengths $m \in \{6, 50\}$. The comparison with Greedy Coordinate Gradient (GCG) (Zou et al., 2023), the standard hard-prompt baseline, is mixed: PPT methods substantially outperform GCG on the reinforced urn at $m = 6$ (PPT-RB reaches the Dyck optimum on every seed) and on reverse cross-entropy on the reinforced urn at $m = 50$, while GCG is competitive or stronger on the Beta–Bernoulli BFT and on Dyck validity for the reinforced urn at $m = 50$. We note that GCG backpropagates through the transformer at every step, unlike PPT and PPT-RB.

Outline. The remainder of the paper is organized as follows. Section 2 establishes the formal setup and connects the elicitation problem to the latent-posterior factorization (2). Section 3 develops PPT and derives the gradient estimator. The experimental results follow in the remainder.

A notation table is included in Appendix A for reference.

2 SETUP

2.1 FORMALIZING THE ELICITATION PROBLEM

Let $\mathcal{Y}$ be a finite vocabulary, $\Delta(\mathcal{Y})$ the simplex of pmfs on $\mathcal{Y}$, and $\mathcal{Y}^* := \bigcup_{n\geq 0} \mathcal{Y}^n$. Throughout we specialize to a trained autoregressive transformer $\pi_\theta : \mathcal{Y}^* \to \Delta(\mathcal{Y})$, $y_{1:n} \mapsto \pi_\theta(\cdot \mid y_{1:n})$, with the empty history $y_{1:0} := \varnothing$. We work in the *prefix-tuning* regime: the prompt $z_{1:m} \in \mathcal{Y}^m$ is the input to the transformer. Fix a continuation horizon $N \geq 1$ and let $Y_{1:N} \in \mathcal{Y}^N$ denote the length-N continuation generated autoregressively from π_θ given $z_{1:m}$, i.e. $Y_t \sim \pi_\theta(\cdot \mid z_{1:m}, Y_{1:t-1})$ for $t = 1, \ldots, N$. We formalize target behavior via a utility function $U : \mathcal{Y}^N \to \mathbb{R}$ evaluated on this continuation, with target behavior gaining higher utility. Given a hard prompt $z_{1:m} \in \mathcal{Y}^m$, the *elicitation objective* is

$$J(z_{1:m}) := \mathbb{E}[U(Y_{1:N})]. \tag{1}$$

Definition 1 (Elicitation problem)**.** *Given a sequence model π_θ, a utility function U, and a prompt length m, the elicitation problem is to find $z_{1:m} \in \mathcal{Y}^m$ maximizing $J(z_{1:m})$.*

2.2 BAYES-FILTERED TRANSFORMERS AND THE LATENT-POSTERIOR FACTORIZATION

Bayes-filtered transformers. A Bayes-filtered transformer (BFT) is one meta-trained on sequences from a two-stage hierarchical process: a latent task $\tilde{k}$ is drawn from a prior Π_0 on a class $\mathcal{K} := \{\tilde{k} : \mathcal{Y}^* \to \Delta(\mathcal{Y})\}$ of one-step predictive kernels, and observations are then drawn conditional on $\tilde{k}$.

Meta-learning a BFT. A BFT π_θ is obtained by minimizing the autoregressive log-loss $\frac{1}{D}\sum_{j=1}^{D}\sum_{i=1}^{n} -\log \pi_\theta(y_i^{(j)} \mid y_{1:i-1}^{(j)})$ over a training set $\{y_{1:n}^{(j)}\}_{j=1}^{D}$. The loss itself is standard for transformer pretraining; what distinguishes BFT training is the data. Each sequence is drawn from the prior predictive: $\tilde{k}^{(j)} \sim \Pi_0$, then $y_{1:n}^{(j)} \sim \tilde{k}^{(j)}$. Crucially, the autoregressive log-loss on such sequences is minimized at every position by the Bayesian posterior predictive distribution (PPD) under Π_0 given the observed prefix (Ortega et al., 2019); this is the sense in which the training targets are *Bayes-filtered*. In the limit of infinite data, infinite capacity, and perfect optimization, the trained transformer realizes this PPD; we call this limit the *idealized BFT*. In practice, with finite data, finite parameters, and finite optimization, the trained transformer only approximates the idealized BFT, and the quality of this approximation has been studied empirically across a range of settings (Mikulik et al., 2020; Genewein et al., 2023; Grau-Moya et al., 2024). Equation (2) below states the idealized (exact) factorization that the trained π_θ inherits asymptotically, for large prefix lengths n, provided the training-induced approximation error is well-behaved.

The latent-posterior factorization. Concretely, when the trained BFT realizes the PPD, as the idealized BFT does, its one-step predictive distribution is a Bayes mixture over the latent kernel:

$$\pi_\theta(y \mid y_{1:n}) = \int_{\mathcal{K}} \tilde{k}(y \mid y_{1:n})\, \Pi(d\tilde{k} \mid y_{1:n}), \qquad y \in \mathcal{Y}, \tag{2}$$

where $\Pi(d\tilde{k} \mid y_{1:n}) \propto \big[\prod_{i=1}^{n} \tilde{k}(y_i \mid y_{1:i-1})\big]\, \Pi_0(d\tilde{k})$ is the posterior under Π_0 given the prefix $y_{1:n}$.

Factoring J through the latent posterior. Let us overload $\tilde{k}$ from a one-step kernel to length-N blocks via the autoregressive product $\tilde{k}(y_{1:N} \mid z_{1:m}) := \prod_{t=1}^{N} \tilde{k}(y_t \mid z_{1:m}, y_{1:t-1})$, i.e., the joint probability that the next N tokens equal $y_{1:N}$ when each is sampled from $\tilde{k}$ given the prefix accumulated so far. Set $\mu(\tilde{k};\, z_{1:m}) := \sum_{y_{1:N}\in\mathcal{Y}^N} U(y_{1:N})\, \tilde{k}(y_{1:N} \mid z_{1:m})$, the expected U over continuations sampled from $\tilde{k}$ given $z_{1:m}$. Under (2), the tower property yields

$$J(z_{1:m}) = \int_{\mathcal{K}} \mu(\tilde{k};\, z_{1:m})\, \Pi(d\tilde{k} \mid z_{1:m}). \tag{3}$$

Thus a prompt $z_{1:m}$ scores high J whenever its posterior $\Pi(\cdot \mid z_{1:m})$ puts much of its mass on latents with high μ (Figure 1).

3 POSTERIOR PREFIX TUNING (PPT)

The factorization (3) expresses $J(z_{1:m})$ as a posterior expectation but leaves the optimization variable $z_{1:m} \in \mathcal{Y}^m$ discrete. Existing hard-prompt methods such as greedy coordinate gradient (GCG)

(Zou et al., 2023) obtain gradient signal in $z_{1:m}$ by backpropagating through the transformer's embedding layer, requiring per-step transformer calls and gradients.

Posterior Prefix Tuning (PPT) instead considers a kernel $\kappa \in \mathcal{K}$ in the same class as the latent (written $z_{1:m} \sim \kappa$) and optimizes a surrogate J_{tilt} in κ. The four objects PPT is built on are summarised in Table 1; full details are in Appendix E.3.

Table 1: The four objects on which PPT is built.

Tilt factor	$W(\tilde{k};\,\kappa) := \mathbb{E}_{z_{1:m}\sim\kappa}\big[\prod_{i=1}^{m} \tilde{k}(z_i \mid z_{1:i-1})\big]$	Tilted posterior	$\Pi_{\mathrm{tilt}}(d\tilde{k};\,\kappa) \propto W(\tilde{k};\,\kappa)\,\Pi_0(d\tilde{k})$
Marginalized utility	$\bar{\mu}(\tilde{k};\,\kappa) := \mathbb{E}_{z_{1:m}\sim\kappa}\big[\mu(\tilde{k};\,z_{1:m})\big]$	Surrogate objective	$J_{\mathrm{tilt}}(\kappa) := \int_{\mathcal{K}} \bar{\mu}(\tilde{k};\,\kappa)\,\Pi_{\mathrm{tilt}}(d\tilde{k};\,\kappa)$

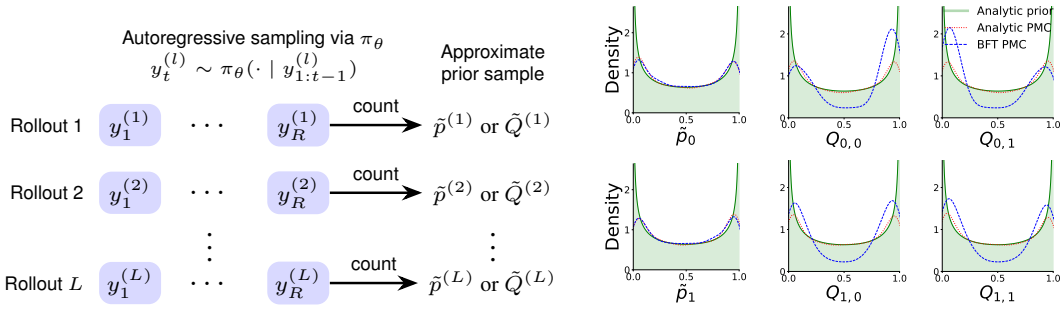
Comparing with (3), J_{tilt} replaces the latent posterior $\Pi(\cdot \mid z_{1:m})$ with the tilted version $\Pi_{\mathrm{tilt}}(\cdot;\,\kappa)$ and $\mu(\tilde{k};\,z_{1:m})$ with the marginalized utility $\bar{\mu}(\tilde{k};\,\kappa)$.

To make the definitions in Table 1 operational we need κ to admit a finite-dimensional parameterization. We obtain this by restricting Π_0 to r-Markov kernels: a kernel $\tilde{k} \in \mathcal{K}$ is *r-Markov* if $\tilde{k}(\cdot \mid h)$ depends on h only through the length-r suffix $\mathrm{suf}_r(h)$, and we write $\mathcal{K}_r \subseteq \mathcal{K}$ for this subset (formal background in Appendix B). When Π_0 is supported on $\mathcal{K}_r$, both the latent and prompt kernels are finite-dimensional: a pmf in $\Delta(\mathcal{Y})$ when $r = 0$, and a transition table in $\mathcal{Q}_r$ when $r \geq 1$. Sections 3.1 and 3.2 instantiate the definitions in Table 1 for $r = 0$ and $r \geq 1$ and develop the IS gradient estimator under each.

The gradient of J_{tilt} with respect to κ's parameter splits into two terms:

$$\nabla J_{\mathrm{tilt}}(\kappa) = \mathrm{Cov}_{\tilde{k}\sim\Pi_{\mathrm{tilt}}(\cdot;\,\kappa)}\big(\bar{\mu}(\tilde{k};\,\kappa),\,\nabla \log W(\tilde{k};\,\kappa)\big) + \mathbb{E}_{\tilde{k}\sim\Pi_{\mathrm{tilt}}(\cdot;\,\kappa)}\big[\nabla \bar{\mu}(\tilde{k};\,\kappa)\big], \tag{4}$$

where all gradients are with respect to κ's parameter. The first term is a covariance under Π_{tilt} between $\bar{\mu}$ and the score of $\log W$; the second vanishes when $\bar{\mu}$ does not depend on κ's parameter. We estimate (4) via importance sampling with samples drawn once from Π_0 via Predictive Monte Carlo (PMC, Appendix C.1; Figure 2a). No further transformer calls or backpropagation through π_θ are needed during optimization.



(a) PMC schematic

(b) Validation, $r = 0$

(c) Validation, $r = 1$

Figure 2: Predictive Monte Carlo (PMC; figure adapted from Ng et al. (2026)). **(a)** Each of L rollouts autoregressively samples a length-R sequence from π_θ with no conditioning; token frequencies give $\tilde{p}^{(l)}$ (0-Markov) or transition counts give $\tilde{Q}^{(l)}$ (r-Markov). **(b)–(c)** PMC validation: green (filled) is the analytic prior density, red (dotted) is from L=5000 rollouts of the exact Bayesian predictive rule, blue (dashed) is from L=5000 rollouts of the trained BFT (R=2000).

3.1 The exchangeable case ($r = 0$)

Consider a BFT trained on i.i.d. sequences: the prior Π_0 is supported on $\mathcal{K}_0$, the i.i.d. kernels of the form $\tilde{k}(y \mid h) = \tilde{p}(y)$ for some pmf $\tilde{p} \in \Delta(\mathcal{Y})$. Identifying $\mathcal{K}_0$ with $\Delta(\mathcal{Y})$, the latent-posterior factorization (2) becomes a Bayes mixture over the latent pmf:

$$\pi_\theta(y \mid y_{1:n}) = \int_{\Delta(\mathcal{Y})} \tilde{p}(y)\,\Pi(d\tilde{p} \mid y_{1:n}), \qquad \Pi(d\tilde{p} \mid y_{1:n}) \propto \Big[\prod_{i=1}^{n} \tilde{p}(y_i)\Big]\,\Pi_0(d\tilde{p}). \tag{5}$$

Instantiating the definitions in Table 1. The prompt kernel $\kappa \in \mathcal{K}_0$ is determined by a pmf $\alpha \in \Delta(\mathcal{Y})$ via $\kappa(y \mid h) = \alpha(y)$. The objects in Table 1 become functions of $\tilde{p}$ and α. The **tilt factor** takes the form

$$W(\tilde{p};\, \alpha) = \mathbb{E}_{z_{1:m} \sim \alpha^{\otimes m}} \left[\prod_{j=1}^{m} \tilde{p}(z_j) \right] = \left[\sum_{v \in \mathcal{Y}} \alpha(v)\, \tilde{p}(v) \right]^m. \tag{6}$$

For $r = 0$, $\mu(\tilde{k};\, z_{1:m})$ does not depend on $z_{1:m}$ (the latent kernel ignores history); abbreviate

$$\mu(\tilde{p}) \;:=\; \mu(\tilde{k};\, z_{1:m}) \;=\; \mathbb{E}_{Y_{1:N} \sim \tilde{p}^{\otimes N}}[U(Y_{1:N})]. \tag{7}$$

The marginalized utility $\bar{\mu}(\tilde{p};\, \alpha) = \mu(\tilde{p})$ trivially.

Score function gradient. The gradient of J_{tilt} specializes (4) (the second term vanishes since $\bar{\mu} = \mu(\tilde{p})$ is α-independent):

$$\nabla_\alpha J_{\mathrm{tilt}}(\alpha) = \mathrm{Cov}_{\tilde{p} \sim \Pi_{\mathrm{tilt}}(\cdot;\, \alpha)} \big(\mu(\tilde{p}),\, \nabla_\alpha \log W(\tilde{p};\, \alpha) \big). \tag{8}$$

Example. In the idealized Beta–Bernoulli BFT, $p := \tilde{p}(1) \sim \mathrm{Beta}(1/2, 1/2)$ is the latent probability of a 1; write $a := \alpha(1)$ for the prompt probability of a 1. Reverse cross-entropy with target $1/2 < \tau^\star < 1$ rewards ones, so $\mu(p)$ increases with p (Appendix D.1). Equation (6) gives $W(p; a) = [ap + (1-a)(1-p)]^m$. Its score $\partial_a \log W$ also increases with p (Appendix C.2), making the covariance in (8) positive: increasing a favors higher-utility latents. Thus J_{tilt} is maximized at $a = 1$, where $W = p^m$ yields the posterior of the optimal all-ones prompt and $J_{\mathrm{tilt}}(1) = J(1, \ldots, 1)$. This equality holds because the prompt is deterministic; it need not hold after snapping an interior a.

IS estimator. We estimate the gradient (8) by importance sampling from the prior. Since Π_0 does not depend on α, we draw samples $\tilde{p}^{(\ell)} \overset{\text{i.i.d.}}{\sim} \Pi_0$, $\ell = 1, \ldots, L$, once and reuse them across all optimization steps; in practice, these samples are obtained via PMC (Appendix C.1).

Given the current α, define the **importance weights** $w_\ell := W(\tilde{p}^{(\ell)};\, \alpha)$ and their self-normalized form $\tilde{w}_\ell := w_\ell / \sum_{j=1}^{L} w_j$, and let $\widehat{\mu}^{(\ell)}$ denote an estimator of $\mu(\tilde{p}^{(\ell)})$. We estimate $\nabla_\alpha J_{\mathrm{tilt}}(\alpha)$ by the self-normalized sample covariance:

$$\widehat{\nabla_\alpha J_{\mathrm{tilt}}}(\alpha) := \sum_{\ell=1}^{L} \tilde{w}_\ell \left(\widehat{\mu}^{(\ell)} - \bar{\mu}_w \right) \left(s^{(\ell)} - \bar{s}_w \right), \tag{9}$$

where $s^{(\ell)} := \nabla_\alpha \log W(\tilde{p}^{(\ell)};\, \alpha)$, $\bar{\mu}_w := \sum_\ell \tilde{w}_\ell \widehat{\mu}^{(\ell)}$, and $\bar{s}_w := \sum_\ell \tilde{w}_\ell s^{(\ell)}$. The two variants of (9) differ in $\widehat{\mu}^{(\ell)}$: PPT-RB uses the closed-form $\mu(\tilde{p}^{(\ell)})$ from (7) when available (Appendix D.3), while PPT uses a single rollout $\widehat{\mu}^{(\ell)} := U(Y^{(\ell)}_{1:N})$ with $Y^{(\ell)}_{1:N} \sim (\tilde{p}^{(\ell)})^{\otimes N}$. The rollout estimator is unbiased given $\tilde{p}^{(\ell)}$; by Rao–Blackwell (Appendix C.3), PPT-RB has lower variance.

3.2 The r-Markov exchangeable case ($r \geq 1$)

This subsection generalizes Section 3.1 to $r \geq 1$, with $r = 0$ recovered as a special case. Consider a BFT trained on r-Markov exchangeable sequences: the prior Π_0 is supported on $\mathcal{K}_r$, the r-Markov kernels of the form $\tilde{k}(y \mid h) = \tilde{Q}_{\mathrm{suf}_r(h),\, y}$ for some transition table $\tilde{Q} \in \mathcal{Q}_r := \Delta(\mathcal{Y})^{|\mathcal{Y}|^r}$. Identifying $\mathcal{K}_r$ with $\mathcal{Q}_r$, the latent-posterior factorization (2) becomes a Bayes mixture over the latent transition table:

$$\pi_\theta(y \mid y_{1:n}) = \int_{\mathcal{Q}_r} \tilde{Q}_{\mathrm{suf}_r(y_{1:n}),\, y}\, \Pi(d\tilde{Q} \mid y_{1:n}), \quad \Pi(d\tilde{Q} \mid y_{1:n}) \propto \prod_{i=r+1}^{n} \tilde{Q}_{\mathrm{suf}_r(y_{1:i-1}),\, y_i} \Pi_0(d\tilde{Q}). \tag{10}$$

Instantiating the definitions in Table 1. The prompt kernel $\kappa \in \mathcal{K}_r$ is determined by a transition table $A \in \mathcal{Q}_r$ via $\kappa(y \mid h) = A_{\mathrm{suf}_r(h),\, y}$, where $A_{s,v}$ is the probability of emitting v from state s, along with an initial state distribution $\rho \in \Delta(\mathcal{Y}^r)$: write $P_{A,\rho}$ for the resulting Markov-chain distribution on $\mathcal{Y}^m$, under which the first r tokens have distribution ρ and each subsequent token z_j

has distribution $A(\cdot \mid \mathrm{suf}_r(z_{1:j-1}))$. The objects in Table 1 become functions of $\tilde{Q}, A, \rho$. The **tilt factor** takes the form

$$W(\tilde{Q};\, A, \rho) = \mathbb{E}_{z_{1:m}\sim P_{A,\rho}}\left[\prod_{j=r+1}^{m} \tilde{Q}_{\mathrm{suf}_r(z_{1:j-1}),\, z_j}\right], \tag{11}$$

where the product starts at $j = r+1$ because the first r tokens have no preceding context to transition from; ρ enters (11) only through $P_{A,\rho}$, which determines the chain's starting state. Appendix C.2 gives a closed-form matrix-power expression for W, which we use throughout.

For r-Markov $\tilde{k}$, $\mu(\tilde{k};\, z_{1:m})$ depends on $z_{1:m}$ only through its length-r suffix; abbreviate

$$\mu(\tilde{Q};\, s) \;:=\; \mu(\tilde{k};\, z_{1:m})\big|_{\mathrm{suf}_r(z_{1:m})=s} \;=\; \mathbb{E}_{Y_{1:N}\sim\tilde{Q}(\cdot|s)}[U(Y_{1:N})], \qquad s \in \mathcal{Y}^r. \tag{12}$$

Closed forms for $\mu(\tilde{Q};\, s)$ in our utility/BFT pairs are in Appendix D.3. The marginalized utility is the average of $\mu(\tilde{Q};\, s)$ over the distribution $\nu_s(A, \rho)$ that the suffix $\mathrm{suf}_r(z_{1:m})$ has under $P_{A,\rho}$:

$$\bar{\mu}(\tilde{Q};\, A, \rho) = \sum_{s\in\mathcal{Y}^r} \nu_s(A, \rho)\, \mu(\tilde{Q};\, s), \tag{13}$$

with $\nu_s(A, \rho)$ given in closed form in Appendix C.2.

Score function gradient. The gradient with respect to A has two terms, one through the tilt weights and one through $\nu_s(A, \rho)$:

$$\nabla_A J_{\mathrm{tilt}}(A, \rho) = \mathrm{Cov}_{\tilde{Q}\sim\Pi_{\mathrm{tilt}}}\big(\bar{\mu}(\tilde{Q};\, A, \rho),\, \nabla_A \log W(\tilde{Q};\, A, \rho)\big) + \mathbb{E}_{\tilde{Q}\sim\Pi_{\mathrm{tilt}}}\big[\nabla_A \bar{\mu}(\tilde{Q};\, A, \rho)\big]. \tag{14}$$

Both $\nabla_A \log W$ and $\nabla_A \bar{\mu}$ reduce to matrix-chain differentiation through closed-form expressions (Appendix C.2). We optimize ρ jointly with A by simplex-projected gradient ascent (Appendix C.5).

The IS estimator extends (9): the second term of (14) contributes an additional importance-weighted expectation, and the integrand $\bar{\mu}(\tilde{Q}^{(\ell)};\, A, \rho)$ can be evaluated either in closed form (PPT-RB) or by a single rollout (PPT). The full estimator is in Appendix C.4.

Recovering a hard prompt. The optimization above returns a kernel κ from which prompts are sampled, but the elicitation problem asks for a hard prompt $z_{1:m} \in \mathcal{Y}^m$. We recover one by snapping the continuous optimum: for $r = 0$ this means rounding the per-token sampling frequencies; for $r \geq 1$, constructing a hard prompt from the optimized transition table via an Eulerian path on its de Bruijn graph. Both procedures are given in Appendix C.6.

4 Experiments

Our experiments use $\mathcal{Y} = \{0, 1\}$, continuation horizon $N = 4$, Markov orders $r \in \{0, 1\}$, and prompt lengths $m \in \{6, 50\}$. We meta-train two BFTs. For the *Beta–Bernoulli BFT* ($r = 0$), $\Pi_0 = \mathrm{Beta}(1/2, 1/2)$ on the latent pmf $\tilde{p}$ and tokens are conditionally i.i.d. $\mathrm{Bernoulli}(\tilde{p})$; for the *reinforced urn BFT* ($r = 1$), Π_0 has independent rows $\tilde{Q}_{a,\cdot} \overset{\text{ind.}}{\sim} \mathrm{Dir}(1/2, 1/2)$ on the latent transition matrix $\tilde{Q} \in \mathcal{Q}_1$, and the sequence is a time-homogeneous Markov chain with kernel $\tilde{Q}$. In both cases the posterior on the latent kernel and the posterior predictive on the next token are available in closed form (Appendices D.1 and D.2); architectures and training details are in Appendix E.2.

4.1 Utility functions

Reverse cross-entropy against a target distribution. Fix a target distribution $\xi^\star$ on $\mathcal{Y}^N$ and set $U_{\xi^\star}(y_{1:N}) := \log \xi^\star(y_{1:N})$. Then $J(z_{1:m})$ is the negative cross-entropy from $\pi_\theta(\cdot \mid z_{1:m})$ to $\xi^\star$, so maximizing J rewards prompts whose continuation distribution concentrates on continuations that $\xi^\star$ assigns high probability. For the Beta–Bernoulli BFT we take $\xi^\star = \mathrm{Bernoulli}(\tau^\star)^{\otimes N}$ and sweep $\tau^\star \in \{0.1, 0.2, 0.3, 0.4, 0.6, 0.7, 0.8, 0.9\}$ (8 targets). For the reinforced urn BFT we take $\xi^\star$ to be the law of a 1-Markov chain with transition matrix $Q^\star$, swept over two families: a *symmetric* family (`sym`$-\tau^\star$) with $Q^\star_{0,0} = Q^\star_{1,1} = \tau^\star$ for $\tau^\star \in \{0.1, 0.2, 0.3, 0.4, 0.6, 0.7, 0.8, 0.9, 1.0\}$ (9 targets), and a *random* family (`dir`$-s$) of 10 matrices indexed by $s \in \{0, \ldots, 9\}$, with rows drawn i.i.d. from

$\mathrm{Dir}(1/2, 1/2)$ under seed s. We omit $\tau^\star = 0.5$ from the Bernoulli and symmetric target sweeps because it gives uniform targets under which $J = -N \log 2$ for every prompt and method.

Match target frequency. Set $U_{q^\star}(y_{1:N}) := -\big(f(y_{1:N}) - q^\star\big)^2$, where $f(y_{1:N}) := \frac{1}{N}\sum_{t=1}^{N} y_t$ is the empirical 1-frequency. We sweep $q^\star \in \{0,\, 0.1, \ldots, 1.0\}$ (11 targets); full specification in Appendix E.1.

Dyck validity. Identifying $0 \equiv$ "(" and $1 \equiv$ ")", set $U_{\mathrm{dyck}}(y_{1:N}) = 1$ if $y_{1:N}$ is a valid balanced-bracket sequence and 0 otherwise; full specification in Appendix E.1.

4.2 Prefix-tuning methods compared

We compare five methods. **GCG** (Zou et al., 2023) optimizes hard prompts $z_{1:m} \in \mathcal{Y}^m$ directly via gradient-guided coordinate substitution (Appendix E.4). **PPT** and **PPT-RB** (Section 3) optimize a continuous parameterization ((A, ρ) or α) of the prompt distribution and snap to a hard prompt at termination (Appendix C.6), with prior samples drawn from the trained BFT via PMC (Appendix C.1); closed-form μ (Appendix D.3) is available for all three utilities under both BFTs, so PPT-RB is applicable throughout. As diagnostic baselines, **PPT (ANALYTIC)** and **PPT-RB (ANALYTIC)** run the same procedure with samples drawn directly from Π_0, isolating the contribution of PMC sampling error. Implementation details are in Appendix E.

Computational cost. During optimization, PPT and PPT-RB never query the BFT, but amortize the computational cost of optimization by first sampling $L = 5000$ rollouts of length $R = 2000$ from the BFT in the course of PMC sampling, meaning a total amortized cost of LR queries to the BFT. The (ANALYTIC) variants never query the transformer. Every step during optimization, GCG queries the transformer $|\mathcal{Y}|^N = 16$ times in the course of evaluating $J(z)$, then backpropagates before using those gradients to select some number $c \leq m$ of bitflips to consider, and querying the transformer $|\mathcal{Y}|^N = 16$ times again to determine $J(z)$ under each of those bitflips, resulting in $O((1+c)|\mathcal{Y}|^N) \leq O((1+m)|\mathcal{Y}|^N)$ queries to the BFT each step.

4.3 Evaluation and results

For any hard prompt $z_{1:m} \in \mathcal{Y}^m$ the elicitation objective $J(z_{1:m}) = \mathbb{E}_{Y_{1:N} \sim \pi_\theta(\cdot|z_{1:m})}[U(Y_{1:N})]$ is computed exactly by enumerating the $|\mathcal{Y}|^N = 16$ continuations through π_θ; this serves as our gold-standard score. For $m = 6$ we additionally enumerate all $|\mathcal{Y}|^m = 64$ candidate prompts to obtain the global optimum $J_{\mathrm{opt}} = \max_{z \in \mathcal{Y}^m} J(z)$ and the rank of every prompt; for $m = 50$ the prompt space has $\sim 10^{15}$ elements, so J_{opt} and rank are unavailable.

Every (utility, prompt length, BFT) setting is run over 10 random seeds and we report means with standard errors. Table 2 summarizes reverse cross-entropy on the reinforced urn BFT. Per-configuration tables for all utilities and BFTs are in Appendix F.1; ESS and PMC diagnostics for PPT and PPT-RB are in Appendix F.2.

Overall, the comparison depends on the utility, BFT, and prompt length. For frequency match, on Beta–Bernoulli all methods reach near-optimum at $m = 6$ and GCG outperforms the PPT variants at $m = 50$; on the reinforced urn, the PPT variants outperform GCG at $m = 6$, and PPT-RB and GCG are comparable at $m = 50$. For Dyck validity, on Beta–Bernoulli all four PPT variants reach the $m = 6$ optimum and all methods produce $J \approx 0.12$ at $m = 50$; on the reinforced urn at $m = 6$, PPT-RB uniquely reaches the enumerated optimum on every seed, and at $m = 50$ the picture inverts with GCG producing $J = 0.61$ versus $J \approx 0.014$–0.021 for the four PPT variants. The remainder of this section focuses on reverse cross-entropy; per-cell breakdowns for frequency match and Dyck validity are in Appendix F.1.

Reverse cross-entropy. On Beta–Bernoulli at $m = 6$, all methods achieve J at the enumerated optimum for every $\tau^\star$. At $m = 50$ on Beta–Bernoulli, GCG and PPT-RB match each other across all $\tau^\star$, while PPT is consistently worse (Table 6). On the reinforced urn, Table 2 reports mean J at both $m \in \{6, 50\}$: the PPT variants achieve J substantially closer to the optimum than GCG on both the symmetric and random target families. At $m = 6$, Table 8 reports the rank of each snapped prompt out of $|\mathcal{Y}|^m = 64$: the PPT variants reach rank 1–3 on most cells while GCG averages rank 5–16.

Table 2: Reverse cross-entropy on the reinforced urn BFT. Each cell is the mean of J over $n = 10$ random seeds (max convention; higher better), with standard error in parentheses. Best method per m in bold (within 10^{-3}).

	$m = 6$					$m = 50$				
	GCG	PPT-RB (A)	PPT-RB	PPT (A)	PPT	GCG	PPT-RB (A)	PPT-RB	PPT (A)	PPT
sym-0.1	−2.51 (0.29)	**−1.75** (0.01)	−1.80 (0.00)	−1.76 (0.01)	−1.80 (0.00)	−1.48 (0.10)	**−0.61** (0.00)	−0.61 (0.00)	−0.62 (0.02)	−0.65 (0.02)
sym-0.2	−2.21 (0.18)	**−1.73** (0.01)	−1.76 (0.00)	**−1.73** (0.01)	−1.76 (0.00)	−1.56 (0.06)	**−1.01** (0.00)	**−1.01** (0.00)	−1.02 (0.01)	−1.03 (0.01)
sym-0.3	−2.23 (0.11)	**−1.94** (0.01)	−1.96 (0.00)	−1.95 (0.01)	−1.96 (0.00)	−1.83 (0.04)	**−1.50** (0.00)	**−1.50** (0.00)	**−1.50** (0.00)	**−1.50** (0.00)
sym-0.4	−2.43 (0.05)	**−2.29** (0.00)	−2.30 (0.00)	−2.29 (0.00)	−2.30 (0.00)	−2.24 (0.02)	**−2.08** (0.00)	−2.22 (0.14)	**−2.08** (0.00)	**−2.08** (0.00)
sym-0.6	−2.37 (0.06)	**−2.18** (0.00)	**−2.18** (0.00)	**−2.18** (0.00)	**−2.18** (0.00)	−2.28 (0.03)	**−2.06** (0.00)	**−2.06** (0.00)	**−2.06** (0.00)	**−2.06** (0.00)
sym-0.7	−2.11 (0.14)	**−1.71** (0.01)	**−1.71** (0.01)	**−1.71** (0.01)	**−1.71** (0.01)	−1.93 (0.06)	**−1.47** (0.00)	**−1.47** (0.00)	**−1.47** (0.00)	**−1.47** (0.00)
sym-0.8	−2.01 (0.22)	**−1.36** (0.01)	**−1.36** (0.01)	**−1.36** (0.01)	**−1.36** (0.01)	−1.71 (0.09)	**−0.96** (0.00)	**−0.96** (0.00)	**−0.96** (0.00)	**−0.96** (0.00)
sym-0.9	−2.19 (0.35)	**−1.17** (0.02)	**−1.17** (0.02)	**−1.17** (0.02)	**−1.17** (0.02)	−1.72 (0.15)	**−0.53** (0.00)	**−0.53** (0.00)	**−0.53** (0.00)	**−0.53** (0.00)
sym-1.0	−24.10 (4.79)	**−10.48** (0.23)	−10.62 (0.22)	**−10.48** (0.23)	−10.62 (0.22)	−17.75 (2.01)	**−1.50** (0.01)	**−1.50** (0.01)	**−1.50** (0.01)	**−1.50** (0.01)
dir-0	−3.47 (0.42)	**−1.89** (0.00)	−2.04 (0.15)	**−1.89** (0.00)	−2.04 (0.15)	−1.76 (0.03)	**−1.70** (0.16)	−1.94 (0.20)	−1.90 (0.18)	−2.01 (0.19)
dir-1	−2.49 (0.18)	**−1.63** (0.00)	**−1.63** (0.00)	**−1.63** (0.00)	**−1.63** (0.00)	−1.65 (0.04)	**−1.30** (0.00)	−1.48 (0.18)	−1.65 (0.23)	−1.66 (0.24)
dir-2	−2.12 (0.18)	**−1.32** (0.00)	**−1.32** (0.00)	**−1.32** (0.00)	**−1.32** (0.00)	−2.32 (0.08)	−1.10 (0.15)	**−0.95** (0.00)	−1.25 (0.20)	−1.26 (0.20)
dir-3	−2.00 (0.20)	**−1.40** (0.04)	**−1.40** (0.04)	**−1.40** (0.04)	**−1.40** (0.04)	−1.74 (0.09)	**−1.02** (0.04)	**−1.02** (0.04)	−1.02 (0.04)	**−1.02** (0.04)
dir-4	−2.93 (0.18)	−2.04 (0.23)	−2.10 (0.37)	−2.04 (0.23)	**−2.04** (0.23)	−2.72 (0.09)	−1.14 (0.37)	**−0.64** (0.27)	−1.61 (0.37)	−1.33 (0.37)
dir-5	−2.27 (0.23)	**−1.75** (0.00)	**−1.75** (0.00)	**−1.75** (0.00)	**−1.75** (0.00)	−1.36 (0.05)	**−0.78** (0.00)	**−0.78** (0.00)	−0.82 (0.02)	−0.80 (0.01)
dir-6	−2.24 (0.20)	−1.45 (0.13)	**−1.37** (0.12)	−1.45 (0.13)	**−1.37** (0.12)	−1.79 (0.15)	−1.19 (0.19)	**−0.93** (0.18)	−1.19 (0.19)	−1.19 (0.19)
dir-7	−2.27 (0.37)	**−1.22** (0.01)	**−1.22** (0.01)	−1.23 (0.01)	**−1.22** (0.01)	−1.92 (0.15)	−0.54 (0.05)	−0.54 (0.05)	−0.55 (0.04)	**−0.52** (0.05)
dir-8	−2.61 (0.13)	**−2.45** (0.03)	**−2.45** (0.03)	**−2.45** (0.03)	**−2.45** (0.03)	**−2.13** (0.02)	−2.21 (0.08)	−2.22 (0.08)	−2.20 (0.07)	−2.21 (0.08)
dir-9	−2.51 (0.19)	**−1.82** (0.00)	−1.93 (0.11)	**−1.82** (0.00)	−1.93 (0.11)	−1.74 (0.03)	**−1.68** (0.12)	**−1.68** (0.12)	−1.80 (0.16)	−1.90 (0.17)

5 Related Works

The Bayesian view of ICL. Our framework rests on the view that ICL admits an approximate Bayesian interpretation, in which the transformer's predictions arise from a posterior over latent predictive models updated by the context. Xie et al. (2022) introduced this view as an explanation of ICL, and Panwar et al. (2024) developed it empirically across a range of synthetic settings. Garg et al. (2022) and Akyürek et al. (2023) validated the picture in in-context linear regression, showing that transformers trained on hierarchical data implement Bayesian predictors. The connection between meta-training on hierarchical data and Bayesian posterior prediction was formalized by Ortega et al. (2019), whose analysis underlies our notion of an idealized BFT.

Belief-state geometry of trained transformers. The latent-posterior factorization (2) is a hypothesis about what the trained transformer represents internally. Shai et al. (2024) showed that transformers trained on hidden Markov processes linearly represent the Bayesian belief state in their residual stream, with geometry matching the mixed-state presentation of computational mechanics. Piotrowski et al. (2025) extended this to constrained belief updates that account for the simplices observed in trained models, and Riechers et al. (2025) showed that next-token pretraining on data with hidden structure implies ICL of that structure. These results indicate that the latent-posterior model is not merely a useful idealization: trained transformers can be observed to maintain and update a posterior over latent generators.

Prefix tuning for BFTs. The closest related works are Genewein et al. (2025) and Wenliang et al. (2026), both of which prompt Beta–Bernoulli BFTs (the $r = 0$ case in our paper) with different Beta hyperparameters than ours: Genewein et al. (2025) use $\alpha = \beta = 1$ (uniform-bias coin pretraining), and Wenliang et al. (2026) sweep over (α, β). Wenliang et al. (2026) additionally consider two simpler Bernoulli BFTs whose latent prior is a point mass at a single coin bias or a two-point mixture. Neither work covers the $r \geq 1$ Markov case.

The prompt/prefix tuning problem as defined in Genewein et al. (2025) refers to aligning the BFT to a *target* sequence distribution $\xi_{\text{Target}}(y_{1:N})$ by *prepending* a learnable prefix $z_{1:m} \in \mathcal{S}^m$, minimizing the forward (mass-seeking) cross-entropy $\mathbb{E}_{y_{1:N} \sim \xi_{\text{Target}}}[-\log \pi_\theta(y_{1:N} \mid z_{1:m})]$ over several prefix alphabets; with our reverse cross-entropy utility, J is the negation of the reverse (mode-seeking) form. Wenliang et al. (2026) share the BFT setting and the forward cross-entropy objective, but their focus is empirical: they show that exhaustive search and intuitive prompts often fail to identify the theoretically optimal prompt, and that optimal prompts depend on the pretraining distribution in unintuitive ways.

Hard-prompt elicitation and persona-modulation attacks. Outside the Bayesian view, a parallel literature attacks the elicitation problem by direct hard-prompt search. Zou et al. (2023) introduced Greedy Coordinate Gradient (GCG), which optimizes a hard prompt to elicit a target completion via gradient-guided coordinate substitution; we use GCG as our experimental baseline. GEPA (Agrawal et al., 2026) instead optimizes instruction text by LLM reflection on execution traces. Both query the model for every candidate (forward and backward passes, or rollouts and reflection calls), whereas

PPT's inner loop makes no model calls after a one-time, utility-independent PMC sampling cost. A complementary line of work exploits the latent structure that PPT operates on directly. Shah et al. (2023) demonstrate that "persona-modulation attacks," designed to steer a model into adopting a particular persona, are an effective and transferable jailbreak strategy, and Deshpande et al. (2023) show that persona assignment systematically shifts a model's distribution of completions. From the latent-posterior viewpoint, both can be understood as engineering prompts that bias the latent posterior toward kernels with high adversarial utility.

Personas as latent factors. A growing body of work treats LLM behavior as conditioned on structured latent character traits, or *personas*. The Persona Selection Model of Marks et al. (2026) hypothesizes that the assistant character is one component of a richer latent posterior maintained by post-trained LLMs, and frames misalignment as undesirable persona selection. Ghandeharioun et al. (2024) similarly frame alignment in terms of personas, arguing that both the model's adopted persona and its inferred user persona shape responses. Joshi et al. (2024) treat personas as latent factors and use this framing to explain the emergence of a "truth direction" in the latent space of language models. Most recently, Wang et al. (2026) exhibit persona features in fine-tuned models that mediate emergent misalignment under narrow fine-tuning. From our perspective, these strands of evidence point to $\mathcal{K}$ as the locus of behavioral steering: elicitation reduces to placing the latent posterior on a kernel, persona, or other latent factor that produces the target behavior.

6 Discussion

Summary. We formalized the elicitation problem for Bayes-filtered transformers and showed that, under the latent-posterior factorization (2), a tilted surrogate J_{tilt} of the elicitation objective admits a gradient (4) estimable entirely in latent-prior space. Posterior Prefix Tuning (PPT) operationalizes this in 0- and 1-Markov exchangeable BFTs: the prompt is sampled from a kernel whose continuous parameter is a pmf on $\mathcal{Y}$ ($r = 0$) or a transition matrix on $\mathcal{Y}$ ($r \geq 1$), and the gradient is estimated by importance sampling with samples drawn from the latent prior. The prior samples are obtained once via predictive Monte Carlo and reused across optimization steps and across utility functions, so each gradient step requires zero transformer calls and no backpropagation through π_θ.

Limitations. The method requires (2) to hold for the trained transformer. Even in our stylized settings, where meta-training data is drawn from a r-Markov exchangeable prior so the idealized log-loss optimum satisfies (2), the trained BFT only approximates this representation; real-world transformers approximate it at best loosely. PPT is currently restricted to the prefix-tuning regime; extending it to settings where the BFT additionally conditions on observed data is left for future work. Our experiments are restricted to $|\mathcal{Y}| = 2$, $r \in \{0, 1\}$, and continuation horizon $N = 4$. PPT may also fail under kernel-class misspecification: an order-0 prompt kernel cannot favor one ordering over another with the same token counts, whereas the reinforced urn posterior depends on transition counts (Appendix G). The surrogate $J_{\mathrm{tilt}}(\kappa)$ generally differs from the expected hard-prompt utility $\mathbb{E}_{z_{1:m}\sim\kappa}[J(z_{1:m})]$, although they coincide when κ is a point mass. This gap has no general sign, and we do not establish a bound on its magnitude. Consequently, improving the surrogate need not improve the expected utility of prompts sampled from κ. Hard-prompt recovery is a separate procedure from the continuous optimization (floor/ceil candidate evaluation for $r = 0$; Eulerian-path construction on the de Bruijn graph for $r = 1$), and we do not establish that the recovered hard prompt is optimal among those consistent with the optimized κ.

Outlook. The L PMC samples are utility-independent and can be reused across any number of utility functions; the cost of approximating Π_0 via PMC is paid once. Extending PPT beyond r-Markov exchangeability to other latent classes with finite-dimensional parameterizations would broaden the regime in which the latent-posterior factorization is operational. PPT also composes naturally with interpretability tools that recover a trained BFT's implicit prior and posterior, since both rely on the same generative view of ICL.

Extending to real-scale LLMs. PMC can be run on a real-scale LLM by generating unconditional continuations and estimating a functional from each rollout, but reading these estimates as latent-prior samples requires the model's predictive rule to converge to a limiting random measure; we do not expect low-order Markov kernels to capture LLM semantics. PPT additionally requires a tractable parametric representation that makes W and its score computable. Establishing these requirements is future work (Appendix G).

AI USE STATEMENT

In this work, we used generative AI tools to aid and polish writing, to draft sections of the paper, for retrieval and discovery (e.g., finding related work), and for research ideation and execution. AI assistants proposed edits and drafted passages of the text, helped locate related work, and assisted with research ideas and with implementing and running experiments. We have reviewed all AI-assisted work: drafted and edited text was revised by the authors and checked against our derivations and code, and suggested references were verified against the cited works. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Code is available at `https://anonymous.4open.science/r/elicitation-4218`. Experimental settings are given in Appendix E, the derivations and closed-form utilities for both BFTs in Appendix D, and the methodology details (predictive Monte Carlo, the tilt factor and its score, and hard-prompt recovery) in Appendix C.

REFERENCES

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alex Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective prompt evolution can outperform reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL `https://openreview.net/forum?id=RQm2KQTM5r`.

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`.

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 (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf`.

Ameet Deshpande, Vishvak Murahari, Tanmay Rajpurohit, Ashwin Kalyan, and Karthik Narasimhan. Toxicity in ChatGPT: Analyzing persona-assigned language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 2023.

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. doi: 10.1093/jrsssb/qkad005.

Sandra Fortini and Sonia Petrone. Exchangeability, Prediction and Predictive Modeling in Bayesian Statistics. *Statistical Science*, 40(1):40 – 67, 2025. doi: 10.1214/24-STS965. URL `https://doi.org/10.1214/24-STS965`. See Theorem 4.6 (Diaconis–Freedman representation) and Examples 4.8–4.9 (reinforced urn scheme).

Shivam Garg, Dimitris Tsipras, Percy Liang, and Gregory Valiant. What can Transformers learn In-Context? A case study of simple function classes. In *Advances in Neural Information Processing Systems*, 2022.

Tim Genewein, Gregoire Deletang, 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 *Proceedings of the 40th International Conference on Machine Learning*, pp. 11173–11195. PMLR, 2023.

Tim Genewein, Li Kevin Wenliang, Jordi Grau-Moya, Anian Ruoss, Laurent Orseau, and Marcus Hutter. Understanding Prompt Tuning and In-Context Learning via Meta-Learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=avRktRfQ8c.

Asma Ghandeharioun, Ann Yuan, Marius Guerard, Emily Reif, Michael A. Lepori, and Lucas Dixon. Who's asking? user personas and the mechanics of latent misalignment. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=eSes1Mic9d.

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 *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=B1ajnQyZgK.

Geoffrey Irving, Joseph Bloom, and Tomek Korbak. Eliciting bad contexts. AI Alignment Forum, January 2025. URL https://www.alignmentforum.org/posts/inkzPmpTFBdXoKLqC/eliciting-bad-contexts.

Nitish Joshi, Javier Rando, Abulhair Saparov, Najoung Kim, and He He. Personas as a way to model truthfulness in language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 6346–6359. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.364. URL https://aclanthology.org/2024.emnlp-main.364/.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 3045–3059, 2021.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 4582–4597, 2021.

Sam Marks, Jack Lindsey, and Christopher Olah. The Persona Selection Model: Why AI assistants might behave like humans. Anthropic Alignment Science Blog, February 2026. URL https://alignment.anthropic.com/2026/psm/.

Vladimir Mikulik, Grégoire Delétang, Tom McGrath, Tim Genewein, Miljan Martic, Shane Legg, and Pedro Ortega. Meta-trained agents implement Bayes-optimal agents. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 18691–18703. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/d902c3ce47124c66ce615d5ad9ba304f-Paper.pdf.

Kenyon Ng, Edwin Fong, David Tyler Frazier, Jeremias Knoblauch, and Susan Wei. TabMGP: Martingale posterior with TabPFN. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=nJ3yX2ZnOJ.

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, 2019. URL https://arxiv.org/abs/1905.03030.

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.

Mateusz Piotrowski, Paul M. Riechers, Daniel Filan, and Adam Shai. Constrained belief updates explain geometric structures in transformer representations. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=f6Hl60FBFU.

Paul M. Riechers, Henry R. Bigelow, Eric A. Alt, and Adam Shai. Next-token pretraining implies in-context learning, 2025. URL https://arxiv.org/abs/2505.18373.

Rusheb Shah, Quentin Feuillade-Montixi, Soroush Pour, Arush Tagade, Stephen Casper, and Javier Rando. Scalable and transferable black-box jailbreaks for language models via persona modulation, 2023. URL https://arxiv.org/abs/2311.03348. Presented at the NeurIPS 2023 Workshop on Socially Responsible Language Modelling Research (SoLaR).

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 (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 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.

Miles Wang, Tom Dupré 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.

Li Kevin Wenliang, Anian Ruoss, Jordi Grau-Moya, Marcus Hutter, and Tim Genewein. Why is prompting hard? Understanding prompts on binary sequence predictors. In Emtiyaz Khan, Yingzhen Li, Arno Solin, and Aaditya Ramdas (eds.), *Proceedings of The 29th International Conference on Artificial Intelligence and Statistics*, volume 300 of *Proceedings of Machine Learning Research*, pp. 2134–2142. PMLR, 02–05 May 2026. URL https://proceedings.mlr.press/v300/wenliang26a.html.

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*, 2022. URL https://openreview.net/forum?id=RdJVFCHjUMI.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models, 2023. URL https://arxiv.org/abs/2307.15043.

A NOTATION

Table 3: Main notation. References link to the defining equation, table, or paragraph. For measurable S, $\Delta(S)$ denotes probability measures on S. Kernel probabilities of sequences use autoregressive products; proportional measures are normalized to mass one. In the Markov rows, $s \in \mathcal{Y}^r$ and $v \in \mathcal{Y}$; the $\mathcal{K}_r$ identity holds for every h with $|h| \geq r$.

Symbol	Definition/type	Meaning	Defined in
$\mathcal{Y}$	Finite set	Token vocabulary.	§ 2.1
m	$\in \mathbb{Z}^+$	Hard prompt length.	§ 2.1
$z_{1:m}$	$\in \mathcal{Y}^m$	Hard prompt.	§ 2.1
N	$\in \mathbb{Z}^+$	Continuation length.	§ 2.1
θ	$\in \mathbb{R}^d$	Transformer weights.	§ 2.1
π_θ	$: \mathcal{Y}^* \to \Delta(\mathcal{Y})$	Transformer's next-token pmf.	§ 2.1
U	$: \mathcal{Y}^N \to \mathbb{R}$	Continuation utility.	§ 2.1
$\Delta(\mathcal{Y})$	$:= \{p \in [0,1]^{\mathcal{Y}} : \sum_{v \in \mathcal{Y}} p(v) = 1\}$	Token probability mass functions.	§ 2.1
$Y_{1:N}$	$\sim \pi_\theta(\cdot \mid z_{1:m})$	Random continuation.	§ 2.1
$\mathcal{Y}^*$	$:= \bigcup_{j=0}^{\infty} \mathcal{Y}^j$	Finite histories (including empty).	§ 2.1
$J(z_{1:m})$	$:= \mathbb{E}_{Y_{1:N} \sim \pi_\theta(\cdot \mid z_{1:m})}[U(Y_{1:N})]$	Expected continuation utility.	Eq. (1)
$\tilde{k}$	$\in \mathcal{K}$	Latent predictive kernel.	§ 2.2
Π_0	$\in \Delta(\mathcal{K})$	Latent prior.	§ 2.2
κ	$\in \mathcal{K}$	Prompt-generating kernel.	§ 3
$\mathcal{K}$	$:= \{\tilde{k} : \mathcal{Y}^* \to \Delta(\mathcal{Y})\}$	All predictive kernels.	§ 2.2
$\Pi(d\tilde{k} \mid z_{1:m})$	$\propto \tilde{k}(z_{1:m}) \Pi_0(d\tilde{k})$	Posterior given $z_{1:m}$.	§ 2.2
$\mu(\tilde{k}; z_{1:m})$	$:= \mathbb{E}_{Y_{1:N} \sim \tilde{k}(\cdot \mid z_{1:m})}[U(Y_{1:N})]$	Expected utility under $\tilde{k}$.	§ 2.2
$\bar{\mu}(\tilde{k}; \kappa)$	$:= \mathbb{E}_{z_{1:m} \sim \kappa}[\mu(\tilde{k}; z_{1:m})]$	Latent utility averaged over prompts.	Table 1
$W(\tilde{k}; \kappa)$	$:= \mathbb{E}_{z_{1:m} \sim \kappa}[\tilde{k}(z_{1:m})]$	Expected prompt likelihood.	Table 1
$\Pi_{\mathrm{tilt}}(d\tilde{k}; \kappa)$	$\propto W(\tilde{k}; \kappa) \Pi_0(d\tilde{k})$	Prior reweighted by W.	Table 1
$J_{\mathrm{tilt}}(\kappa)$	$:= \mathbb{E}_{\tilde{k} \sim \Pi_{\mathrm{tilt}}(\cdot; \kappa)}[\bar{\mu}(\tilde{k}; \kappa)]$	PPT surrogate objective.	Table 1
r	$\in \mathbb{Z}_{\geq 0}$	History length used by a kernel.	§ 3
$\mathcal{K}_r$	$:= \{q \in \mathcal{K} : q(\cdot \mid h) = q(\cdot \mid \mathrm{suf}_r(h))\}$	Last-r history dependence.	§ 3
$\mathrm{suf}_r(h)$	$:= h_{\lvert h \rvert - r + 1 : \lvert h \rvert}$	Final r tokens ($\lvert h \rvert \geq r$).	§ 3
$\tilde{p}(v)$	$:= \tilde{k}(v \mid \varnothing)$	Latent token pmf ($r = 0$).	§ 3.1
$\alpha(v)$	$:= \kappa(v \mid \varnothing)$	Prompt token pmf ($r = 0$).	§ 3.1
$\mathcal{Q}_r$	$:= \Delta(\mathcal{Y})^{\lvert \mathcal{Y} \rvert^r}$	Space of transition tables.	§ 3.2
$\tilde{Q}_{s,v}$	$:= \tilde{k}(v \mid s)$	Latent transition table ($r \geq 1$).	§ 3.2
$A_{s,v}$	$:= \kappa(v \mid s)$	Prompt transition table ($r \geq 1$).	§ 3.2
$\rho(s)$	$:= \Pr_{z_{1:m} \sim \kappa}(z_{1:r} = s)$	Initial state distribution.	§ 3.2
$P_{A,\rho}(z_{1:m})$	$:= \rho(z_{1:r}) \prod_{j=r+1}^{m} A_{z_{j-r:j-1}, z_j}$	Prompt distribution ($m \geq r$).	§ 3.2
$\nu_s(A, \rho)$	$:= \Pr_{z_{1:m} \sim P_{A,\rho}}(\mathrm{suf}_r(z_{1:m}) = s)$	Probability of final suffix s.	§ 3.2
L	$\in \mathbb{Z}^+$	Number of PMC rollouts.	§ C.1
R	$\in \mathbb{Z}^+$	Tokens per PMC rollout.	§ C.1
w_ℓ	$:= W(\tilde{k}^{(\ell)}; \kappa)$	Unnormalized importance weight.	§ 3.1
$\tilde{w}_\ell$	$:= w_\ell / \sum_{j=1}^{L} w_j$	Normalized importance weight.	§ 3.1

B MARKOV EXCHANGEABILITY

The representation (2) is deliberately broad: it posits that the model's one-step predictive rule can be written as a Bayes mixture over latent one-step predictors $\tilde{k} \in \mathcal{K}$. On its own, however, this

representation is not yet an algorithmic tool for the inference-time objectives in Section 1: the space $\mathcal{K}$ contains arbitrary history-dependent kernels, so the posterior $\Pi(\cdot \mid y_{1:n})$ is a posterior over an essentially infinite-dimensional object, and evaluating $\mu(\tilde{k})$ may be no easier than rolling out π_θ itself.

To make (2) operational, we restrict attention to structured subclasses in which the latent predictive object admits a *finite-dimensional parameterization* and posterior updating depends on *low-dimensional sufficient statistics* of the observed history (typically token or r-gram transition counts). Exchangeability ($r = 0$) and r-Markov exchangeability provide exactly this structure: by classical representation theorems, such processes are mixtures of i.i.d. models or mixtures of rth-order Markov chains, so the latent object reduces to a random pmf (exchangeable case) or a random transition table (Markov-exchangeable case). In the remainder of this section we recall these notions and introduce the corresponding definitions of *in-context exchangeable* and *in-context r-Markov exchangeable* transformers, which form the stylized setting used throughout the paper.

B.1 Exchangeability

A process $(Y_n)_{n\geq 1}$ on $\mathcal{Y}$ with joint law $\mathbb{P}$ is *exchangeable* if for every n and every permutation σ of $\{1, \dots, n\}$,

$$(Y_1, \dots, Y_n) \overset{d}{=} (Y_{\sigma(1)}, \dots, Y_{\sigma(n)}).$$

By the de Finetti representation theorem, exchangeability is equivalent to the existence of a random pmf $\tilde{p} \in \Delta(\mathcal{Y})$ such that, conditional on $\tilde{p}$, the tokens are i.i.d. with law $\tilde{p}$. Writing Π_0 for the law of $\tilde{p}$, the one-step predictive distribution is

$$\mathbb{P}(Y_{n+1} \in A \mid Y_{1:n} = y_{1:n}) = \int_{\Delta(\mathcal{Y})} \tilde{p}(A)\, \Pi(d\tilde{p} \mid y_{1:n}), \qquad A \subseteq \mathcal{Y},$$

where $\Pi(\cdot \mid y_{1:n})$ is the posterior induced by the likelihood $\prod_{i=1}^{n} \tilde{p}(y_i)$. This is a special case of (2) obtained by restricting Π_0 to the 0-Markov subclass $\mathcal{K}_0$, i.e. kernels of the form $\tilde{k}(\cdot \mid h) \equiv \tilde{p}(\cdot)$.

By Ionescu–Tulcea, the one-step conditionals of π_θ determine a process law $\mathbb{P}_\theta$ on $\mathcal{Y}^{\mathbb{N}}$, on which de Finetti can be invoked. We shall say the transformer π_θ is **in-context exchangeable**, or equivalently **in-context 0-Markov exchangeable**, if $\mathbb{P}_\theta$ is exchangeable.

B.2 1-Markov exchangeability

Let $(Y_n)_{n\geq 1}$ be a stochastic process taking values in a finite vocabulary $\mathcal{Y}$, with joint law $\mathbb{P}$. For a path $y_{1:n}$ define transition counts

$$T^{(n)}_{a,b}(y_{1:n}) := \sum_{t=1}^{n-1} \mathbf{1}\{y_t = a,\ y_{t+1} = b\}, \qquad a, b \in \mathcal{Y}.$$

The process is *Markov exchangeable* if for every n and any two paths $y_{1:n}, y'_{1:n}$,

$$y_1 = y'_1 \text{ and } T^{(n)}_{a,b}(y_{1:n}) = T^{(n)}_{a,b}(y'_{1:n})\ \forall a, b \implies \mathbb{P}(Y_{1:n} = y_{1:n}) = \mathbb{P}(Y_{1:n} = y'_{1:n}).$$

Let

$$\mathcal{Q} := \Big\{ Q = (Q_{a,b})_{a,b\in\mathcal{Y}} :\ Q_{a,b} \in [0,1],\ \sum_{b\in\mathcal{Y}} Q_{a,b} = 1\ \ \forall a \in \mathcal{Y} \Big\}$$

denote the set of row-stochastic transition matrices on $\mathcal{Y}$. Under a recurrence condition, Markov exchangeability is equivalent to the existence of a random transition matrix $\tilde{Q} \in \mathcal{Q}$ such that, conditional on $\tilde{Q}$, (Y_t) is a time-homogeneous Markov chain with transition matrix $\tilde{Q}$ (Diaconis–Freedman; Fortini–Petrone).

For $Q \in \mathcal{Q}$, $a \in \mathcal{Y}$, and $A \subseteq \mathcal{Y}$, write $Q_a(A) := \sum_{b\in A} Q_{a,b}$. Given an observed path $y_{1:n}$, the Markov chain likelihood of Q is $L_n(Q; y_{1:n}) := \prod_{t=1}^{n-1} Q_{y_t, y_{t+1}}$, and the posterior on Q is

$$\Pi^{\mathcal{Q}}(dQ \mid y_{1:n})\ \propto\ L_n(Q; y_{1:n})\, \Pi^{\mathcal{Q}}_0(dQ),$$

where $\Pi_0^{\mathcal{Q}}$ is the prior law of $\tilde{Q}$. Then the one-step predictive distribution is the posterior mixture

$$\mathbb{P}(Y_{n+1} \in A \mid Y_{1:n} = y_{1:n}) = \int_{\mathcal{Q}} Q_{y_n}(A)\, \Pi^{\mathcal{Q}}(dQ \mid y_{1:n}), \qquad A \subseteq \mathcal{Y}. \tag{15}$$

This is a special case of (2) under the 1-Markov restriction $\tilde{k}(y \mid h) = \tilde{Q}_{h_{\text{last}},y}$, i.e. when the latent kernel depends on the history only through its last token. Thus we shall say the transformer π_θ is **in-context 1-Markov exchangeable** if the joint law it induces through its in-context predictions $\mathbb{P}_\theta$ is 1-Markov exchangeable.

B.3 r-Markov exchangeability

Fix $r \geq 1$ and define the r-gram (block) process $S_t := (Y_{t-r+1}, \ldots, Y_t) \in \mathcal{Y}^r$ for $t \geq r$. We call (Y_t) *r-Markov exchangeable* if the block process $(S_t)_{t \geq r}$ is Markov exchangeable in the sense of Section B.2 (with state space $\mathcal{Y}^r$). Under the corresponding recurrence condition, the Diaconis–Freedman representation applied to (S_t) yields a mixture of 1-step Markov chains on $\mathcal{Y}^r$, equivalently a mixture of rth-order Markov chains on $\mathcal{Y}$.

Equivalently, (2) holds with Π_0 supported on $\mathcal{K}_r$, i.e. $\tilde{k}(\cdot \mid h)$ depends on h only through $\mathrm{suf}_r(h)$. Thus we shall say the transformer π_θ is **in-context r-Markov exchangeable** if the joint law it induces through its in-context predictions $\mathbb{P}_\theta$ is r-Markov exchangeable.

C Methodology details

C.1 Predictive Monte Carlo

Given any sequence model π_θ, *Predictive Monte Carlo* (PMC) Fortini & Petrone (2025); Fong et al. (2023) produces approximate samples from the latent prior Π_0 that π_θ implies, using only forward passes through the model, with no weights, gradients, or explicit access to Π_0. An idealized BFT realizes the corresponding Π_0 by construction; a trained BFT only approximately satisfies (2), and PMC samples inherit the approximation error of π_θ relative to the idealized BFT. PPT (and PPT-RB) require samples from Π_0 over the latent predictive object ($\tilde{p}$ for 0-Markov exchangeable, $\tilde{Q}$ for r-Markov exchangeable), and we use PMC to obtain them.

Procedure. For each rollout $l = 1, \ldots, L$:

1. Autoregressively sample a length-R sequence from π_θ with no conditioning:
$$y_t^{(l)} \sim \pi_\theta(\cdot \mid y_{1:t-1}^{(l)}), \qquad t = 1, \ldots, R.$$
2. Extract a sample of the latent predictive object by counting:
   - 0-*Markov exchangeable*: count the token frequencies to obtain $\tilde{p}^{(l)}(y) = \frac{\#\{t : y_t^{(l)} = y,\ 1 \leq t \leq R\}}{R}$ for each $y \in \mathcal{Y}$.
   - *r-Markov exchangeable*: count the empirical transition frequencies from $y_{1:R}^{(l)}$ and normalize to obtain $\tilde{Q}^{(l)}$.

Under (2), as $R \to \infty$ these empirical frequencies converge almost surely to the latent $\tilde{p}$ or $\tilde{Q}$ Fortini & Petrone (2025), so the counts yield an (approximate) sample from Π_0.

Figure 2a illustrates the procedure. The key feature is that PMC requires only *forward passes* through π_θ (no gradients, no internal access), making it applicable even with API-only access to the model.

C.2 Closed-form computation of W and its score

Exchangeable case. The tilt factor (6) is $W(\tilde{p};\, \alpha) = [\sum_v \alpha(v)\tilde{p}(v)]^m$. For $\mathcal{Y} = \{0, 1\}$ the score with respect to α has the closed form

$$\nabla_\alpha \log W(\tilde{p};\, \alpha) = \frac{m\,[\tilde{p} - (1 - \tilde{p})]}{\alpha\tilde{p} + (1 - \alpha)(1 - \tilde{p})}.$$

Markov-exchangeable case. The tilt factor (11) is an expectation over the prompt Markov chain on $\mathcal{Y}^m$. Changing variables from token sequences to state sequences gives a closed-form matrix-power expression:

$$W(\tilde{Q};\, A, \rho) = \rho^\top \big(M^{(\tilde{Q})}\big)^{m-r}\, \mathbf{1}, \tag{16}$$

where $M^{(\tilde{Q})} \in \mathbb{R}^{|\mathcal{Y}|^r \times |\mathcal{Y}|^r}$ has entries

$$M^{(\tilde{Q})}_{s,s'} = \sum_{v \in \mathcal{Y}} A_{s,v}\, \tilde{Q}_{s,v}\, \mathbf{1}[\mathrm{suf}_r(s,v) = s']. \tag{17}$$

That is, $M^{(\tilde{Q})} = A \odot \tilde{Q}$ in the Hadamard sense, after accounting for the suffix map. The cost is $O((m-r)\cdot|\mathcal{Y}|^{r+1}\cdot L)$ across all L prior samples. When $r = 0$, M is 1×1 with $M = \sum_v \alpha(v)\tilde{p}(v)$, ρ is trivial, and (16) recovers (6).

Gradient $\nabla_A \log W$. Differentiating (16) through the matrix power gives

$$\nabla_A \log W(\tilde{Q};\, A, \rho) = \frac{\nabla_A \big[\rho^\top (M^{(\tilde{Q})})^{m-r}\, \mathbf{1}\big]}{\rho^\top (M^{(\tilde{Q})})^{m-r}\, \mathbf{1}},$$

with $\nabla_A M^{(\tilde{Q})}_{s,s'} = \tilde{Q}_{s,v}\, \mathbf{1}[\mathrm{suf}_r(s,v) = s']$ for the corresponding $A_{s,v}$ entry. The matrix-chain derivative is computed by differentiating through the matrix power.

Suffix distribution $\nu_s(A,\rho)$. The distribution that $\mathrm{suf}_r(z_{1:m})$ has under $P_{A,\rho}$ is given by

$$\nu_s(A,\rho) = [\rho^\top T_A^{m-r}]_s,$$

where $T_A \in \mathbb{R}^{|\mathcal{Y}|^r \times |\mathcal{Y}|^r}$ is the state transition matrix induced by A, with entries $[T_A]_{s,s'} = \sum_v A_{s,v}\, \mathbf{1}[\mathrm{suf}_r(s,v) = s']$. The gradient $\nabla_A \bar{\mu}$ in (14) is computed by differentiating $\rho^\top T_A^{m-r}$ through T_A.

C.3 Rao–Blackwellization of the gradient estimator

For each prior sample $\tilde{k}^{(\ell)} \sim \Pi_0$ the IS estimator requires an estimate of the integrand $\bar{\mu}(\tilde{k}^{(\ell)};\, \kappa)$ (and, for $r \geq 1$, of $\nabla\bar{\mu}$). PPT-RB uses the closed-form $\bar{\mu}$; PPT uses a single Monte Carlo rollout: sample $z_{1:m} \sim \kappa$ and $Y_{1:N} \sim \tilde{k}^{(\ell)}(\cdot \mid \mathrm{suf}_r(z_{1:m}))$, then set $\widehat{\bar{\mu}}^{(\ell)} := U(Y_{1:N})$.

Variance reduction. The rollout estimate is unbiased given $\tilde{k}^{(\ell)}$:

$$\mathbb{E}\Big[\widehat{\bar{\mu}}^{(\ell)} \,\Big|\, \tilde{k}^{(\ell)}\Big] = \bar{\mu}(\tilde{k}^{(\ell)};\, \kappa).$$

The law of total variance gives

$$\mathrm{Var}\Big(\widehat{\bar{\mu}}^{(\ell)}\Big) \;=\; \mathrm{Var}\Big(\bar{\mu}(\tilde{k}^{(\ell)};\, \kappa)\Big) + \mathbb{E}\Big[\mathrm{Var}\Big(\widehat{\bar{\mu}}^{(\ell)} \,\Big|\, \tilde{k}^{(\ell)}\Big)\Big], \tag{18}$$

so $\mathrm{Var}(\bar{\mu}(\tilde{k}^{(\ell)};\, \kappa)) \;\leq\; \mathrm{Var}(\widehat{\bar{\mu}}^{(\ell)})$, with the gap equal to the within-kernel sampling variance $\mathbb{E}[\mathrm{Var}(\widehat{\bar{\mu}}^{(\ell)} \mid \tilde{k}^{(\ell)})]$. Substituting the closed-form $\bar{\mu}$ for $\widehat{\bar{\mu}}^{(\ell)}$ in the IS estimator ((9) for $r = 0$, (19) for $r \geq 1$) reduces the variance of each summand, hence the variance of the gradient estimator. The reduction is largest when individual rollouts from a fixed $\tilde{k}^{(\ell)}$ are noisy.

Specialization to $r = 0$. The kernel ignores history, so $\bar{\mu}(\tilde{p};\, \alpha) = \mu(\tilde{p})$ trivially and the rollout estimate is $\widehat{\mu}^{(\ell)} = U(Y^{(\ell)}_{1:N})$ with $Y^{(\ell)}_{1:N} \sim (\tilde{p}^{(\ell)})^{\otimes N}$ (no prompt sampling needed); PPT-RB uses the closed-form $\mu(\tilde{p}^{(\ell)})$.

When the closed form exists. Closed-form expressions for $\mu(\tilde{p})$ and $\bar{\mu}(\tilde{Q};\, A, \rho)$ depend on the utility/BFT combination; see Appendices D.1 and D.2 for the cases used in our experiments. For utilities where no closed form is available (e.g., a black-box behavioral classifier), only PPT applies.

C.4 IS GRADIENT ESTIMATOR FOR $r \geq 1$

For $r \geq 1$ the score function gradient (14) has two terms; we estimate it by importance sampling using i.i.d. draws $\tilde{Q}^{(\ell)} \overset{\text{i.i.d.}}{\sim} \Pi_0$, $\ell = 1, \ldots, L$, drawn once via PMC (Appendix C.1) and reused across optimization steps. Given the current (A, ρ), define the importance weights

$$w_\ell := W(\tilde{Q}^{(\ell)};\, A, \rho), \qquad \tilde{w}_\ell := \frac{w_\ell}{\sum_{j=1}^{L} w_j}.$$

The estimator is a self-normalized sample covariance plus an importance-weighted expectation:

$$\widehat{\nabla_A J_{\text{tilt}}}(A, \rho) := \sum_{\ell=1}^{L} \tilde{w}_\ell \left(\widehat{\bar{\mu}}^{(\ell)} - \bar{\bar{\mu}}_w\right)\left(s^{(\ell)} - \bar{s}_w\right) \;+\; \sum_{\ell=1}^{L} \tilde{w}_\ell \, \nabla_A \bar{\mu}(\tilde{Q}^{(\ell)};\, A, \rho), \tag{19}$$

where $s^{(\ell)} := \nabla_A \log W(\tilde{Q}^{(\ell)};\, A, \rho)$, $\bar{\bar{\mu}}_w := \sum_\ell \tilde{w}_\ell \widehat{\bar{\mu}}^{(\ell)}$, and $\bar{s}_w := \sum_\ell \tilde{w}_\ell s^{(\ell)}$. The integrand $\bar{\mu}(\tilde{Q}^{(\ell)};\, A, \rho)$ defines the variant: PPT-RB uses the closed-form (13), while PPT uses a single rollout $\widehat{\bar{\mu}}^{(\ell)} = U(Y_{1:N}^{(\ell)})$ with $z^{(\ell)} \sim P_{A,\rho}$ and $Y_{1:N}^{(\ell)} \sim \tilde{Q}^{(\ell)}(\cdot \mid \text{suf}_r(z^{(\ell)}))$. The latter is unbiased: $\mathbb{E}[\widehat{\bar{\mu}}^{(\ell)} \mid \tilde{Q}^{(\ell)}] = \bar{\mu}(\tilde{Q}^{(\ell)};\, A, \rho)$. By Rao–Blackwell (Appendix C.3), PPT-RB has strictly lower variance for each ℓ.

C.5 UPDATING THE INITIAL-STATE DISTRIBUTION ρ

We optimize $\rho \in \Delta(\mathcal{Y}^r)$ jointly with A by projected gradient ascent. Since $J_{\text{tilt}}(A, \rho) = \sum_s \rho_s \left[(M^{(\cdot)})^{m-r}\mathbf{1}\right]_s$ is linear in ρ for fixed A, the gradient is

$$\nabla_\rho J_{\text{tilt}}(A, \rho) = \left(T_A^{m-r}\right)^\top \boldsymbol{\mu}_w,$$

where $[\boldsymbol{\mu}_w]_s := \sum_\ell \tilde{w}_\ell \, \mu(\tilde{Q}^{(\ell)};\, s)$ is the importance-weighted utility at ending state s. Each step, we take $\rho \leftarrow \text{proj}_\Delta\big(\rho + \eta\,(\nabla_\rho J_{\text{tilt}} - \overline{\nabla_\rho J_{\text{tilt}}})\big)$, where $\overline{(\cdot)}$ denotes the mean over components and $\text{proj}_\Delta(x)_s := \max(x_s,\, \epsilon) \big/ \sum_{s'} \max(x_{s'},\, \epsilon)$ with $\epsilon = 10^{-6}$.

C.6 HARD PROMPT RECOVERY

After PPT optimization, we recover a hard prompt $z_{1:m} \in \mathcal{Y}^m$ from the optimized prompt-distribution parameter as follows.

Exchangeable case ($r = 0$). For binary $\mathcal{Y} = \{0, 1\}$, evaluate J_{tilt} at $\alpha(1) = h/m$ for both $h = \lfloor m\alpha(1) \rfloor$ and $h = \lceil m\alpha(1) \rceil$, select the count with higher J_{tilt}, and construct the prompt with that many 1s.

Markov-exchangeable case ($r \geq 1$). Convert the optimized (A, ρ) to a hard prompt $z_{1:m}$ via an Eulerian path in the de Bruijn graph. For each of the two starting states s_0:

1. Compute expected visit counts $\nu(u) = \sum_{j=0}^{m-r-1} [e_{s_0}^\top T_A^j]_u$ for each r-gram state u.
2. Assign edge multiplicities via Hamilton's method (largest-remainder rounding): each row u receives $\hat{\nu}(u) = \text{round}(\nu(u))$ total edges (adjusted to sum to $m-r$), distributed across tokens as $c(u, v) = \text{round}(\hat{\nu}(u) \cdot A_u(v))$ (adjusted to sum to $\hat{\nu}(u)$).
3. If an Eulerian path from s_0 exists on c, find it via Hierholzer's algorithm. Otherwise, try $O(|\mathcal{Y}|^{r+1})$ single-edge swap variants $(c(u, v) \mathrel{-}= 1,\ c(u, v') \mathrel{+}= 1)$ that admit an Eulerian path.
4. Among all candidates admitting an Eulerian path, select the one maximizing J_{tilt}. The edge labels give tokens $z_{r+1:m}$; prepend the r tokens encoding s_0 to recover $z_{1:m}$.

D BFTs: DETAILED DERIVATIONS

D.1 BETA–BERNOULLI

This appendix records the 0-Markov exchangeable (Beta–Bernoulli) specialization of the framework. Write $p := \tilde{p}(1) \in [0,1]$.

Posterior. The posterior predictive given context $y_{1:n}$ is

$$P(Y_{n+1} = 1 \mid y_{1:n}) = \mathbb{E}[\tilde{p} \mid y_{1:n}] = \frac{\alpha + S_1(n)}{\alpha + \beta + n}, \qquad S_1(n) = \sum_{i=1}^{n} y_i,$$

and the posterior on $\tilde{p}$ is $\Pi(\tilde{p} \mid y_{1:n}) = \mathrm{Beta}\big(\alpha + S_1(n),\ \beta + n - S_1(n)\big)$. We use $\alpha = \beta = 1/2$.

Reverse cross-entropy: closed form for J. The reverse cross-entropy utility $U_{\tau^\star}(y_{1:N}) = S_1 \log \tau^\star + S_0 \log(1-\tau^\star)$ is linear in the token counts $S_1 = \sum_{t=1}^{N} y_t$ and $S_0 = N - S_1$. Define the average marginal one-rate $\bar{p}_N(z_{1:m}) := \frac{1}{N}\sum_{t=1}^{N} \Pr(Y_{n+t} = 1 \mid y_{1:n}, z_{1:m})$. Since $\mathbb{E}[S_1] = N\bar{p}_N(z_{1:m})$,

$$J(z_{1:m}) = N\Big[\log(1-\tau^\star) + \bar{p}_N(z_{1:m})\, \log\frac{\tau^\star}{1-\tau^\star}\Big]. \tag{20}$$

This closed form relies on the linearity of $U_{\tau^\star}$ in S_1. The frequency match and Dyck utilities do not enjoy this property and are computed by enumeration over $\mathcal{Y}^N$.

Rao–Blackwellized utilities $\mu(\tilde{p})$. Under $\tilde{p}^{\otimes N}$, the continuation tokens $Y_{n+1}, \ldots, Y_{n+N}$ are i.i.d. $\mathrm{Bernoulli}(p)$.

Reverse cross-entropy ($U_{\tau^\star}$). By the same linearity argument,

$$\mu(\tilde{p}) = N\Big(\log(1-\tau^\star) + p\, \log\frac{\tau^\star}{1-\tau^\star}\Big). \tag{21}$$

Match target frequency ($U_{q^\star}$). The empirical frequency $f = \frac{1}{N}\sum_{t=1}^{N} Y_{n+t}$ has $\mathbb{E}[f] = p$ and $\mathrm{Var}(f) = p(1-p)/N$ under $\tilde{p}^{\otimes N}$. Since $\mathbb{E}[(f-q^\star)^2] = \mathrm{Var}(f) + (\mathbb{E}[f] - q^\star)^2$,

$$\mu(\tilde{p}) = -\Big[\frac{p(1-p)}{N} + (p - q^\star)^2\Big].$$

Dyck validity (U_{dyck}). With $N = 4$, the two valid Dyck sequences $\{0101,\ 0011\}$ each have probability $p^2(1-p)^2$ under $\tilde{p}^{\otimes 4}$ (two zeros and two ones in each). Hence

$$\mu(\tilde{p}) = 2\, p^2(1-p)^2.$$

D.2 REINFORCED URN

This appendix records the 1-Markov exchangeable (reinforced urn) specialization of the framework, with $r = 1$ and $|\mathcal{Y}| = 2$.

Predictive and posterior. By the Diaconis–Freedman representation theorem Fortini & Petrone (2025), the induced predictive rule takes the *reinforced urn* form

$$P(Y_{n+1} = y \mid y_{1:n}) = \frac{\alpha/|\mathcal{Y}| + T_{y_n,y}(y_{1:n})}{\alpha + \sum_{y' \in \mathcal{Y}} T_{y_n,y'}(y_{1:n})},$$

where $T_{a,b}(y_{1:n}) := \sum_{t=1}^{n-1} \mathbf{1}\{y_t = a,\ y_{t+1} = b\}$ counts observed transitions. The posterior on $\tilde{Q}$ has independent Dirichlet rows

$$\Pi(\tilde{Q}_{a,\cdot} \mid y_{1:n}) = \mathrm{Dir}\big(\alpha/|\mathcal{Y}| + T_{a,b}(y_{1:n})\big)_{b \in \mathcal{Y}}, \qquad a \in \mathcal{Y}.$$

We use $\alpha = 1$.

Reverse cross-entropy: closed form for J. The reverse cross-entropy utility $U_{Q^\star}(y_{1:N}) = \sum_{t=0}^{N-1} \log Q^\star_{s_t, y_{n+t+1}}$ is a sum of per-step terms. By linearity of expectation,

$$J(z_{1:m}) = \sum_{t=0}^{N-1} \sum_{u \in \mathcal{Y}} \sum_{y \in \mathcal{Y}} P(s_t = u,\, Y_{n+t+1} = y \mid y_{1:n},\, z_{1:m})\, \log Q^\star_{u,y},$$

where the joint marginals $P(s_t = u,\, Y_{n+t+1} = y \mid y_{1:n},\, z_{1:m})$ can be computed from the model's one-step predictives without enumerating all $|\mathcal{Y}|^N$ sequences. The frequency match and Dyck utilities do not decompose as per-step sums and are computed by enumeration.

Rao–Blackwellized utilities $\mu(\tilde{Q};\, s)$. Conditional on $\tilde{Q}$ and starting state $s \in \mathcal{Y}$, the continuation $Y_{1:N}$ is a Markov chain on $\mathcal{Y}$.

Reverse cross-entropy ($U_{Q^\star}$). The utility decomposes over the N-step context visitation:

$$\mu(\tilde{Q};\, s) = \sum_{t=0}^{N-1} \sum_{u \in \mathcal{Y}} \left[e_s^\top \tilde{Q}^t\right]_u h(u, \tilde{Q}),$$

where $h(u, \tilde{Q}) := \sum_{y \in \mathcal{Y}} \tilde{Q}_{u,y} \log Q^\star_{u,y}$ is the per-state negative cross-entropy and $e_s^\top \tilde{Q}^t$ is the visitation probability over states at step t (obtained by iterating the $|\mathcal{Y}| \times |\mathcal{Y}|$ transition matrix).

Match target frequency ($U_{q^\star}$). Let $p_t := \left[e_s^\top \tilde{Q}^t\right]_1$. Then $\mathbb{E}_{\tilde{Q}}[f] = \frac{1}{N} \sum_{t=1}^{N} p_t$ and

$$\mathrm{Var}_{\tilde{Q}}(f) = \frac{1}{N^2} \sum_{i,j} \mathrm{Cov}_{\tilde{Q}}(Y_i, Y_j),$$

where the covariances follow from the Markov property. Hence

$$\mu(\tilde{Q};\, s) = -\Big[\, \mathrm{Var}_{\tilde{Q}}(f) + \big(\mathbb{E}_{\tilde{Q}}[f] - q^\star\big)^2 \Big].$$

Dyck validity (U_{dyck}). With $N = 4$, the two valid Dyck sequences $\{0101,\, 0011\}$ have Markov chain probabilities:

$$\mu(\tilde{Q};\, s) = \tilde{Q}_{s,0}\, \tilde{Q}_{0,1}\, \tilde{Q}_{1,0}\, \tilde{Q}_{0,1} \;+\; \tilde{Q}_{s,0}\, \tilde{Q}_{0,0}\, \tilde{Q}_{0,1}\, \tilde{Q}_{1,1}.$$

D.3 Summary tables

Table 4: Exact objective $J(z_{1:m})$ for each BFT/utility pair. We write $\bar{p}_N(z_{1:m}) := \frac{1}{N} \sum_{t=1}^{N} \Pr(Y_t = 1 \mid z_{1:m})$ for the average marginal one-rate.

Utility	Beta–Bernoulli	Reinforced urn
Rev. xent.	$N\Big[\log(1-\tau^\star) + \bar{p}_N(z_{1:m}) \log \frac{\tau^\star}{1-\tau^\star}\Big]$	$\sum_{t,u,y} P(s_t = u, Y_{t+1} = y) \log Q^\star_{u,y}$
Freq. match	enumerate	enumerate
Dyck	$\pi_\theta(\{0101, 0011\} \mid z_{1:m})$	$\pi_\theta(\{0101, 0011\} \mid z_{1:m})$

The reverse cross-entropy admits a closed-form J for both BFTs because U is a sum of per-step terms; the other utilities require enumeration over $\mathcal{Y}^N$. All three μ closed forms hold in both BFTs: Beta–Bernoulli benefits from i.i.d. continuation tokens; the reinforced urn uses Markov-chain marginals via $\tilde{Q}^t$ for the reverse cross-entropy and frequency-match utilities, and direct enumeration of the two valid Dyck sequences for the Dyck utility.

Table 5: Closed-form Rao–Blackwellized utility μ for each setting. Here $p := \tilde{p}(1)$, $h(u, \tilde{Q}) := \sum_y \tilde{Q}_{u,y} \log Q^\star_{u,y}$, and $P(s_t = u)$ denotes the context-visitation probability under $\tilde{Q}$ starting from state s.

Utility	Beta–Bernoulli: $\mu(\tilde{p})$	Reinforced urn: $\mu(\tilde{Q};\, s)$
Rev. xent.	$N\Big(\log(1-\tau^\star) + p \log \frac{\tau^\star}{1-\tau^\star}\Big)$	$\sum_{t=0}^{N-1} \sum_u P(s_t = u)\, h(u, \tilde{Q})$
Freq. match	$-\Big[\frac{p(1-p)}{N} + (p - q^\star)^2\Big]$	$-\Big[\mathrm{Var}_{\tilde{Q}}(f) + (\mathbb{E}_{\tilde{Q}}[f] - q^\star)^2\Big]$
Dyck	$2\,p^2(1-p)^2$	$\tilde{Q}_{s,0}\tilde{Q}_{0,1}\tilde{Q}_{1,0}\tilde{Q}_{0,1} + \tilde{Q}_{s,0}\tilde{Q}_{0,0}\tilde{Q}_{0,1}\tilde{Q}_{1,1}$

E Experiments details

E.1 Utility functions

The reverse cross-entropy utility is fully specified in Section 4.1. We give the corresponding details for the frequency-match and Dyck utilities here.

Match target frequency. Fix a target frequency $q^\star \in [0, 1]$ and let $f(y_{1:N}) := \frac{1}{N}\sum_{t=1}^{N} y_t$ be the empirical frequency of token 1 in the continuation. Set $U_{q^\star}(y_{1:N}) := -\big(f(y_{1:N}) - q^\star\big)^2$, the negative squared deviation of the continuation frequency from the target. Maximizing J pulls the continuation frequency toward $q^\star$ in mean square; the utility depends on $y_{1:N}$ only through its count of 1s. We sweep $q^\star \in \{0,\, 0.1,\, 0.2,\, \ldots,\, 1.0\}$ (11 targets).

Dyck validity. Identify 0 with "(" and 1 with ")". A sequence $y_{1:N} \in \mathcal{Y}^N$ is a *valid Dyck* sequence if every prefix contains at least as many "(" as ")" and the full sequence is balanced. We set $U_{\mathrm{dyck}}(y_{1:N}) = 1$ when $y_{1:N}$ is a valid Dyck sequence and 0 otherwise. With $N = 4$ the valid set is $\{0101,\, 0011\}$ out of $|\mathcal{Y}|^N = 16$ continuations, so $J = \mathbb{E}\left[U_{\mathrm{dyck}}\right] = \pi_\theta(\text{valid Dyck} \mid z_{1:m})$.

E.2 BFT training

Beta–Bernoulli transformer. 1-layer pre-norm transformer with $d_{\mathrm{model}} = 64$, 4 attention heads ($d_{\mathrm{head}} = 16$), and feedforward dimension 128. Token embeddings $\phi\colon \mathcal{Y} \cup \{\mathrm{BOS}\} \to \mathbb{R}^{64}$ are learned (3 embeddings: tokens 0, 1, and BOS); no positional encoding is used. Every training and inference sequence is prepended with a BOS token. Each training batch consists of 128 sequences of length 2000, each generated by sampling $\tilde{p} \sim \mathrm{Beta}(1/2, 1/2)$ and generating an i.i.d. $\mathrm{Bernoulli}(\tilde{p})$ sequence. The model is trained for 50,000 iterations with AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay 0.1), cosine learning rate schedule from 3×10^{-4} to 10^{-5} with 500 warmup steps, bfloat16 mixed precision, and gradient clipping at norm 1.0.

Reinforced urn transformer. 8-layer pre-norm transformer with $d_{\mathrm{model}} = 256$, 8 attention heads ($d_{\mathrm{head}} = 32$), and feedforward dimension 1024. Token embeddings $\phi\colon \mathcal{Y} \to \mathbb{R}^{256}$ and position embeddings $\psi\colon \{0, \ldots, 19999\} \to \mathbb{R}^{256}$ are both learned; at each position t the input is $\phi(y_t) + \psi(t)$. A BOS token is prepended, with its own learned embedding. Each training batch consists of 64 sequences of length 2000, each generated by sampling $\tilde{Q}_{u,\cdot} \overset{\text{i.i.d.}}{\sim} \mathrm{Dir}(1/2, 1/2)$ for $u \in \{0, 1\}$ and generating a 1-Markov chain from $\tilde{Q}$ with a random initial token. The model is trained for 100,000 iterations with AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay 0.1), cosine learning rate schedule from 3×10^{-4} to 10^{-5} with 1,000 warmup steps, bfloat16 mixed precision, and gradient clipping at norm 1.0.

E.3 PPT AND PPT-RB

The four PPT variants share the parameterization, optimizer, and IS gradient estimator below; they differ in (a) how prior samples are obtained and (b) how $\widehat{\mu}^{(\ell)}$ ($r = 0$) or $\widehat{\widehat{\mu}}^{(\ell)}$ ($r \geq 1$) in (9) or (19) is computed.

Parameterization and optimizer. For $r = 0$, $\alpha \in \Delta^1$ is parameterized by unconstrained logits $(a_0, a_1) \in \mathbb{R}^2$ via softmax. For $r \geq 1$, A and ρ are optimized jointly by simplex-projected gradient ascent with learning rate $\eta = 0.1$. For A, each row $A_{s,\cdot}$ is updated independently:

$$A_{s,\cdot} \leftarrow \mathrm{proj}_{\Delta}\Big(A_{s,\cdot} + \eta\left(\nabla_{A_{s,\cdot}} J_{\mathrm{tilt}} - \overline{\nabla_{A_{s,\cdot}} J_{\mathrm{tilt}}}\right)\Big),$$

where $\overline{(\cdot)}$ denotes the mean over the $|\mathcal{Y}|$ components and $\mathrm{proj}_{\Delta}(x)_v := \max(x_v, \epsilon) \big/ \sum_{v'} \max(x_{v'}, \epsilon)$ with $\epsilon = 10^{-6}$; ρ is updated identically (Appendix C.5). The gradient $\nabla_A J_{\mathrm{tilt}}$ is computed by differentiating through the matrix power of $M^{(\tilde{Q})}$ (Appendix C.2). Both A and ρ are initialized by independent draws from $\mathrm{Dir}(1, \ldots, 1)$. Optimization terminates early if J_{tilt} does not improve by more than 10^{-5} for 100 consecutive steps.

Drawing samples from Π_0. For PPT and PPT-RB, $L = 5{,}000$ samples are obtained by Predictive Monte Carlo (Appendix C.1) using rollouts of length $R = 2{,}000$ from the trained BFT π_θ. For the (ANALYTIC) variants, $L = 5{,}000$ samples are drawn directly from Π_0: $\tilde{p} \sim \mathrm{Beta}(1/2, 1/2)$ for the Beta–Bernoulli BFT and rows $\tilde{Q}_{a,\cdot} \overset{\text{i.i.d.}}{\sim} \mathrm{Dir}(1/2, 1/2)$ for the reinforced urn BFT. In both cases the L samples are drawn once at the start and reused across optimization steps.

Runtime. The full experiment sweep completes in approximately 8 hours on a single NVIDIA H100 GPU.

Computing $\widehat{\mu}^{(\ell)}$ and $\widehat{\widehat{\mu}}^{(\ell)}$. PPT-RB uses the closed-form $\mu(\tilde{p}^{(\ell)})$ ($r = 0$) or $\bar{\mu}(\tilde{Q}^{(\ell)}; A, \rho)$ ($r \geq 1$) from Appendix D.3. PPT uses a single rollout per prior sample: for $r = 0$ we draw $Y^{(\ell)}_{1:N} \sim (\tilde{p}^{(\ell)})^{\otimes N}$ and set $\widehat{\mu}^{(\ell)} = U(Y^{(\ell)}_{1:N})$; for $r \geq 1$ we draw a prompt $z^{(\ell)} \sim P_{A,\rho}$, then a continuation $Y^{(\ell)}_{1:N} \sim \tilde{Q}^{(\ell)}(\cdot \mid \mathrm{suf}_r(z^{(\ell)}))$, and set $\widehat{\widehat{\mu}}^{(\ell)} = U(Y^{(\ell)}_{1:N})$.

E.4 GCG

Each iteration, if $z^{(t)}_{1:m}$ is the prompt from iteration t, the GCG algorithm evaluates the gradient

$$\nabla_{e_{z_i^{(t)}}} J(z^{(t)}_{1:m}) \in \mathbb{R}^{|\mathcal{Y}|} \tag{22}$$

where $e_{z_i^{(t)}}$ denotes the one-hot vector representing the value of the ith token, and uses this to select the top-k most promising token substitutions $\mathcal{X}_i = \text{Top-}k(\nabla_{e_{z_i^{(t)}}} J(z^{(t)}_{1:m}))$ for each $i \in \{1, \ldots, m\}$.

In our binary setting $|\mathcal{Y}| = 2$, so any $k \geq 2$ leaves $\mathcal{X}_i = \{0, 1\}$ and the gradient ranking has no effect. We therefore set $k = 1$, so $\mathcal{X}_i = \{c_i^{(t)}\}$ where $c_i^{(t)} \in \arg\max_v \nabla_{e_{z_i^{(t)}}, v} J(z^{(t)}_{1:m})$ is the gradient's Top-1 choice at position i. Let $S^{(t)} = \{i : c_i^{(t)} \neq z_i^{(t)}\}$ be the positions where the Top-1 token differs from the current one, and let $z^{(t,i)}$ denote $z^{(t)}_{1:m}$ with position i replaced by $c_i^{(t)}$. The candidate set at iteration t is $\{z^{(t,i)} : i \in S^{(t)}\}$; we update by

$$z^{(t+1)}_{1:m} = \arg\max_{i \in S^{(t)}} J(z^{(t,i)}),$$

setting $z^{(t+1)}_{1:m} = z^{(t)}_{1:m}$ when $S^{(t)} = \emptyset$ or when no candidate improves J. Initialization is $z_i^{(0)} \overset{\text{i.i.d.}}{\sim} \mathrm{Uniform}(\{0, 1\})$. Each iteration costs one backprop through π_θ to obtain the gradient, plus $|S^{(t)}| \cdot |\mathcal{Y}|^N$ forwards (at most $m \cdot |\mathcal{Y}|^N$) to evaluate the candidates. We halt at $t > 10$ for $m = 6$ or $t > 100$ for $m = 50$, or when no candidate improves J.

F EXPERIMENTAL RESULTS

F.1 METHOD COMPARISON

This subsection reports per-configuration performance for the five methods, averaged over 10 random seeds. The convention is that J is to be *maximised*; for $m = 6$, where the full set of $V^m = 64$ hard prompts is enumerable, we additionally report rank-out-of-64 of the snapped prompt (rank 1 = best). Each cell averages J (or rank) over $n = 10$ random seeds, with standard error in parentheses; the best method per row is in bold (within 10^{-3} for J, within 0.05 for rank). Reverse cross-entropy on the reinforced urn at both $m \in \{6, 50\}$ is the main-text Table 2; the corresponding rank table at $m = 6$ is Table 8. Reverse cross-entropy on the Beta–Bernoulli BFT is in Tables 6 and 7.

Table 6: Reverse cross-entropy on the Beta–Bernoulli BFT. Each cell is the mean of J over $n = 10$ random seeds (max convention; higher better), with standard error in parentheses. Best method per m in bold (within 10^{-3}).

	$m = 6$					$m = 50$				
	GCG	PPT-RB (A)	PPT-RB	PPT (A)	PPT	GCG	PPT-RB (A)	PPT-RB	PPT (A)	PPT
$\tau^\star = 0.1$	**−1.06** (0.00)	**−1.06** (0.00)	**−1.06** (0.00)	**−1.06** (0.00)	**−1.06** (0.00)	**−0.51** (0.00)	**−0.51** (0.00)	**−0.51** (0.00)	−1.16 (0.18)	−1.23 (0.20)
$\tau^\star = 0.2$	**−1.29** (0.00)	**−1.29** (0.00)	**−1.29** (0.00)	**−1.29** (0.00)	**−1.29** (0.00)	**−0.95** (0.00)	**−0.95** (0.00)	**−0.95** (0.00)	−1.36 (0.12)	−1.40 (0.13)
$\tau^\star = 0.3$	**−1.67** (0.00)	**−1.67** (0.00)	**−1.67** (0.00)	**−1.67** (0.00)	**−1.67** (0.00)	**−1.46** (0.00)	**−1.46** (0.00)	**−1.46** (0.00)	−1.71 (0.07)	−1.74 (0.08)
$\tau^\star = 0.4$	**−2.16** (0.00)	**−2.16** (0.00)	**−2.16** (0.00)	**−2.16** (0.00)	**−2.16** (0.00)	**−2.06** (0.00)	**−2.06** (0.00)	**−2.06** (0.00)	−2.18 (0.03)	−2.19 (0.04)
$\tau^\star = 0.6$	**−2.16** (0.00)	**−2.16** (0.00)	**−2.16** (0.00)	**−2.16** (0.00)	**−2.16** (0.00)	**−2.06** (0.00)	**−2.06** (0.00)	**−2.06** (0.00)	−2.17 (0.03)	−2.18 (0.04)
$\tau^\star = 0.7$	**−1.67** (0.00)	**−1.67** (0.00)	**−1.67** (0.00)	**−1.67** (0.00)	**−1.67** (0.00)	**−1.46** (0.00)	**−1.46** (0.00)	**−1.46** (0.00)	−1.69 (0.07)	−1.71 (0.08)
$\tau^\star = 0.8$	**−1.29** (0.00)	**−1.29** (0.00)	**−1.29** (0.00)	**−1.29** (0.00)	**−1.29** (0.00)	**−0.95** (0.00)	**−0.95** (0.00)	**−0.95** (0.00)	−1.32 (0.12)	−1.36 (0.13)
$\tau^\star = 0.9$	**−1.05** (0.00)	**−1.05** (0.00)	**−1.05** (0.00)	**−1.05** (0.00)	**−1.05** (0.00)	**−0.51** (0.00)	**−0.51** (0.00)	**−0.51** (0.00)	−1.10 (0.19)	−1.17 (0.21)

Table 7: Reverse cross-entropy, Beta–Bernoulli, $m = 6$. Rank-out-of-64 of the snapped prompt (1 = enumerated optimum), averaged over $n = 10$ random seeds; standard error in parentheses. Best method per row in bold (within 0.05).

	GCG	PPT-RB (A)	PPT-RB	PPT (A)	PPT
$\tau^\star = 0.1$	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
$\tau^\star = 0.2$	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
$\tau^\star = 0.3$	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
$\tau^\star = 0.4$	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
$\tau^\star = 0.6$	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
$\tau^\star = 0.7$	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
$\tau^\star = 0.8$	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
$\tau^\star = 0.9$	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)

Frequency match. On Beta–Bernoulli at $m = 6$ (Tables 9 and 10), all methods reach rank 1 for most $q^\star$ values. At $q^\star \in \{0.6, 0.7\}$, the PPT variants land on suboptimal prompts (rank 6–11) while GCG reaches rank 1–4; at $q^\star \in \{0.3, 0.4\}$ the situation reverses. On the reinforced urn at $m = 6$, the PPT variants substantially outperform GCG at the extremes ($q^\star \in \{0.0, 0.1, 0.2, 0.8, 0.9, 1.0\}$: PPT reaches rank 1–4 while GCG ranks 15–18 of 64), are roughly tied for $q^\star \in \{0.4, 0.5, 0.6\}$, and all methods rank substantially below the optimum at $q^\star = 0.7$. At $m = 50$ on Beta–Bernoulli, GCG achieves J closer to zero than the PPT variants in every cell; on the reinforced urn at $m = 50$, GCG and PPT-RB are roughly competitive, with PPT (without Rao–Blackwellisation) consistently lagging.

Dyck validity. On Beta–Bernoulli at $m = 6$, all four PPT variants reach the enumerated optimum on every seed; GCG averages a worse rank but is within 10^{-4} of the optimal J. At $m = 50$ on Beta–Bernoulli, all methods achieve $J \approx 0.12$. On the reinforced urn at $m = 6$ (Tables 11 and 12), PPT-RB reaches the enumerated optimum on every seed (rank 1 throughout), PPT is close ($J = 0.54$ versus the optimum $J = 0.59$), and the remaining methods (PPT (ANALYTIC), PPT-RB (ANALYTIC), GCG) produce J between 0.29 and 0.38. At $m = 50$ on the reinforced urn the picture inverts: GCG reaches $J = 0.61$ while all four PPT variants produce J between 0.014 and 0.021. The final ESS/L for PPT-RB on this configuration is ≈ 0.07 (mean over 10 seeds), down from ≈ 0.22 at $m = 6$.

Table 8: Reverse cross-entropy on the reinforced urn BFT, $m = 6$. Rank-out-of-64 of the snapped prompt (1 = enumerated optimum), averaged over $n = 10$ random seeds; standard error in parentheses. Best method per row in bold (within 0.05).

	GCG	PPT-RB (A)	PPT-RB	PPT (A)	PPT
`sym-0.1`	4.9 (1.76)	**1.4** (0.16)	2.0 (0.00)	1.5 (0.17)	2.0 (0.00)
`sym-0.2`	4.9 (1.76)	**1.4** (0.16)	2.0 (0.00)	**1.4** (0.16)	2.0 (0.00)
`sym-0.3`	4.9 (1.76)	**1.5** (0.17)	2.0 (0.00)	1.6 (0.16)	2.0 (0.00)
`sym-0.4`	4.9 (1.76)	**1.5** (0.17)	2.0 (0.00)	1.6 (0.16)	2.0 (0.00)
`sym-0.6`	11.0 (3.52)	**1.3** (0.15)	**1.3** (0.15)	**1.3** (0.15)	**1.3** (0.15)
`sym-0.7`	11.0 (3.52)	**1.3** (0.15)	**1.3** (0.15)	**1.3** (0.15)	**1.3** (0.15)
`sym-0.8`	11.0 (3.52)	**1.3** (0.15)	**1.3** (0.15)	**1.3** (0.15)	**1.3** (0.15)
`sym-0.9`	11.0 (3.52)	**1.3** (0.15)	**1.3** (0.15)	**1.3** (0.15)	**1.3** (0.15)
`sym-1.0`	11.0 (3.52)	**1.5** (0.17)	1.6 (0.16)	**1.5** (0.17)	1.6 (0.16)
`dir-0`	16.1 (3.90)	**1.0** (0.00)	2.4 (1.40)	**1.0** (0.00)	2.4 (1.40)
`dir-1`	12.7 (3.68)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
`dir-2`	9.3 (2.68)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
`dir-3`	10.5 (3.33)	**1.6** (0.31)	**1.6** (0.31)	**1.6** (0.31)	**1.6** (0.31)
`dir-4`	7.0 (1.33)	2.9 (0.82)	4.4 (2.57)	2.9 (0.82)	**2.8** (0.76)
`dir-5`	5.0 (1.59)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
`dir-6`	10.9 (3.07)	3.0 (0.82)	**2.5** (0.76)	3.0 (0.82)	**2.5** (0.76)
`dir-7`	11.3 (3.54)	**1.6** (0.16)	**1.6** (0.16)	1.7 (0.15)	**1.6** (0.16)
`dir-8`	10.8 (5.12)	**5.6** (2.40)	**5.6** (2.40)	**5.6** (2.40)	**5.6** (2.40)
`dir-9`	13.5 (4.36)	**1.0** (0.00)	3.5 (2.50)	**1.0** (0.00)	3.5 (2.50)

Table 9: Frequency match. Each cell is the mean of J over $n = 10$ random seeds (max convention; higher better), with standard error in parentheses. Best method per (r, m) in bold (within 10^{-3}).

	$m = 6$					$m = 50$				
	GCG	PPT-RB (A)	PPT-RB	PPT (A)	PPT	GCG	PPT-RB (A)	PPT-RB	PPT (A)	PPT
Beta–Bernoulli ($r = 0$)										
$q^\star = 0.0$	**−0.03** (0.00)	**−0.03** (0.00)	**−0.03** (0.00)	**−0.03** (0.00)	**−0.03** (0.00)	**−0.00** (0.00)	−0.01 (0.00)	−0.01 (0.00)	−0.04 (0.01)	−0.04 (0.01)
$q^\star = 0.1$	**−0.02** (0.00)	**−0.02** (0.00)	**−0.02** (0.00)	**−0.02** (0.00)	**−0.02** (0.00)	**−0.01** (0.00)	−0.02 (0.00)	−0.02 (0.00)	−0.03 (0.01)	−0.03 (0.01)
$q^\star = 0.2$	**−0.04** (0.00)	**−0.04** (0.00)	**−0.04** (0.00)	**−0.04** (0.00)	**−0.04** (0.00)	**−0.03** (0.00)	−0.07 (0.01)	−0.07 (0.01)	−0.07 (0.01)	−0.07 (0.01)
$q^\star = 0.3$	**−0.07** (0.00)	**−0.07** (0.00)	**−0.07** (0.00)	**−0.07** (0.00)	**−0.07** (0.00)	**−0.05** (0.00)	−0.06 (0.01)	−0.06 (0.01)	−0.06 (0.01)	−0.06 (0.01)
$q^\star = 0.4$	**−0.08** (0.00)	**−0.08** (0.00)	**−0.08** (0.00)	**−0.08** (0.00)	**−0.08** (0.00)	**−0.06** (0.00)	−0.07 (0.00)	−0.07 (0.00)	−0.07 (0.00)	−0.07 (0.00)
$q^\star = 0.5$	**−0.09** (0.00)	**−0.09** (0.00)	**−0.09** (0.00)	**−0.09** (0.00)	**−0.09** (0.00)	**−0.07** (0.00)	**−0.07** (0.00)	**−0.07** (0.00)	**−0.07** (0.00)	**−0.07** (0.00)
$q^\star = 0.6$	**−0.08** (0.00)	**−0.08** (0.00)	**−0.08** (0.00)	**−0.08** (0.00)	**−0.08** (0.00)	**−0.06** (0.00)	−0.07 (0.00)	−0.07 (0.00)	−0.07 (0.00)	−0.07 (0.00)
$q^\star = 0.7$	**−0.07** (0.00)	**−0.07** (0.00)	**−0.07** (0.00)	**−0.07** (0.00)	**−0.07** (0.00)	**−0.05** (0.00)	−0.07 (0.01)	−0.07 (0.01)	−0.07 (0.01)	−0.06 (0.01)
$q^\star = 0.8$	**−0.04** (0.00)	**−0.04** (0.00)	**−0.04** (0.00)	**−0.04** (0.00)	**−0.04** (0.00)	**−0.03** (0.00)	−0.08 (0.01)	−0.08 (0.01)	−0.08 (0.01)	−0.08 (0.01)
$q^\star = 0.9$	**−0.02** (0.00)	**−0.02** (0.00)	**−0.02** (0.00)	**−0.02** (0.00)	**−0.02** (0.00)	**−0.01** (0.00)	−0.02 (0.00)	−0.02 (0.00)	−0.03 (0.01)	−0.03 (0.01)
$q^\star = 1.0$	**−0.03** (0.00)	**−0.03** (0.00)	**−0.03** (0.00)	**−0.03** (0.00)	**−0.03** (0.00)	**−0.00** (0.00)	−0.01 (0.00)	−0.01 (0.00)	−0.04 (0.01)	−0.04 (0.01)
Reinforced urn ($r = 1$)										
$q^\star = 0.0$	−0.19 (0.04)	**−0.07** (0.02)	**−0.07** (0.02)	**−0.07** (0.02)	**−0.07** (0.02)	−0.03 (0.00)	**−0.00** (0.00)	−0.01 (0.01)	−0.06 (0.04)	−0.08 (0.04)
$q^\star = 0.1$	−0.13 (0.03)	**−0.05** (0.02)	**−0.05** (0.02)	**−0.05** (0.02)	**−0.05** (0.02)	**−0.02** (0.00)	−0.02 (0.01)	−0.03 (0.01)	−0.06 (0.02)	−0.06 (0.02)
$q^\star = 0.2$	−0.10 (0.02)	**−0.05** (0.01)	**−0.05** (0.01)	**−0.05** (0.01)	**−0.05** (0.01)	**−0.03** (0.00)	−0.05 (0.01)	−0.05 (0.01)	−0.06 (0.01)	−0.07 (0.01)
$q^\star = 0.3$	−0.07 (0.01)	**−0.06** (0.00)	−0.07 (0.00)	**−0.06** (0.00)	−0.07 (0.00)	**−0.03** (0.00)	−0.05 (0.01)	−0.05 (0.01)	−0.06 (0.01)	−0.06 (0.01)
$q^\star = 0.4$	**−0.03** (0.00)	**−0.03** (0.00)	**−0.03** (0.00)	**−0.03** (0.00)	**−0.03** (0.00)	−0.02 (0.00)	**−0.01** (0.00)	**−0.01** (0.00)	−0.04 (0.01)	−0.04 (0.01)
$q^\star = 0.5$	−0.03 (0.00)	**−0.02** (0.00)	−0.02 (0.00)	**−0.02** (0.00)	−0.02 (0.00)	−0.01 (0.00)	**−0.00** (0.00)	**−0.00** (0.00)	−0.02 (0.00)	−0.02 (0.00)
$q^\star = 0.6$	**−0.04** (0.00)	**−0.04** (0.01)	−0.05 (0.01)	**−0.04** (0.01)	−0.05 (0.01)	**−0.01** (0.00)	−0.04 (0.02)	−0.04 (0.02)	−0.06 (0.02)	−0.05 (0.02)
$q^\star = 0.7$	−0.08 (0.02)	**−0.07** (0.00)	−0.07 (0.00)	**−0.07** (0.00)	−0.07 (0.00)	**−0.02** (0.00)	−0.06 (0.02)	−0.06 (0.02)	−0.07 (0.01)	−0.07 (0.01)
$q^\star = 0.8$	−0.11 (0.02)	**−0.05** (0.00)	−0.05 (0.01)	**−0.05** (0.00)	−0.05 (0.01)	**−0.03** (0.00)	−0.05 (0.01)	−0.06 (0.01)	−0.06 (0.01)	−0.07 (0.01)
$q^\star = 0.9$	−0.14 (0.03)	**−0.03** (0.00)	**−0.03** (0.00)	**−0.03** (0.00)	**−0.03** (0.00)	**−0.03** (0.00)	−0.04 (0.01)	−0.07 (0.02)	−0.06 (0.02)	−0.09 (0.02)
$q^\star = 1.0$	−0.20 (0.03)	**−0.04** (0.00)	**−0.04** (0.00)	**−0.04** (0.00)	**−0.04** (0.00)	−0.06 (0.01)	**−0.03** (0.02)	**−0.03** (0.02)	−0.08 (0.04)	−0.09 (0.04)

Table 10: Frequency match, $m = 6$. Rank-out-of-64 of the snapped prompt (1 = enumerated optimum), averaged over $n = 10$ random seeds; standard error in parentheses. Best method per row in bold (within 0.05). Prompts with J within 10^{-5} are assigned the same rank.

	GCG	PPT-RB (A)	PPT-RB	PPT (A)	PPT
Beta–Bernoulli ($r = 0$)					
$q^\star = 0.0$	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
$q^\star = 0.1$	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
$q^\star = 0.2$	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
$q^\star = 0.3$	1.5 (0.50)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
$q^\star = 0.4$	2.0 (1.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
$q^\star = 0.5$	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
$q^\star = 0.6$	**4.0** (1.53)	11.0 (0.00)	11.0 (0.00)	11.0 (0.00)	11.0 (0.00)
$q^\star = 0.7$	**1.0** (0.00)	6.0 (0.00)	6.0 (0.00)	6.0 (0.00)	6.0 (0.00)
$q^\star = 0.8$	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
$q^\star = 0.9$	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
$q^\star = 1.0$	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
Reinforced urn ($r = 1$)					
$q^\star = 0.0$	16.0 (4.91)	**3.9** (2.90)	**3.9** (2.90)	**3.9** (2.90)	**3.9** (2.90)
$q^\star = 0.1$	15.3 (5.08)	**3.9** (2.90)	**3.9** (2.90)	**3.9** (2.90)	**3.9** (2.90)
$q^\star = 0.2$	15.5 (5.63)	**3.1** (2.10)	**3.1** (2.10)	**3.1** (2.10)	**3.1** (2.10)
$q^\star = 0.3$	10.1 (3.87)	**8.3** (2.10)	13.4 (0.93)	**8.3** (2.10)	13.4 (0.93)
$q^\star = 0.4$	**2.0** (0.30)	**2.0** (0.00)	**2.0** (0.00)	**2.0** (0.00)	**2.0** (0.00)
$q^\star = 0.5$	2.2 (0.76)	**1.4** (0.16)	2.0 (0.00)	1.5 (0.17)	2.0 (0.00)
$q^\star = 0.6$	4.7 (1.14)	**4.5** (2.50)	7.0 (3.33)	**4.5** (2.50)	7.0 (3.33)
$q^\star = 0.7$	13.2 (4.73)	**11.6** (2.37)	14.4 (1.58)	**11.6** (2.37)	14.4 (1.58)
$q^\star = 0.8$	17.7 (4.52)	**1.0** (0.00)	3.1 (2.10)	**1.0** (0.00)	3.1 (2.10)
$q^\star = 0.9$	16.7 (4.11)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
$q^\star = 1.0$	16.5 (4.17)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)

Table 11: Dyck validity. Each cell is the mean of J over $n = 10$ random seeds (max convention; higher better), with standard error in parentheses. Best method per row in bold (within 10^{-3}).

BFT	m	GCG	PPT-RB (A)	PPT-RB	PPT (A)	PPT
Beta–Bernoulli ($r = 0$)	6	**+0.10** (0.00)	**+0.10** (0.00)	**+0.10** (0.00)	**+0.10** (0.00)	**+0.10** (0.00)
Beta–Bernoulli ($r = 0$)	50	**+0.12** (0.00)	**+0.12** (0.00)	**+0.12** (0.00)	**+0.12** (0.00)	**+0.12** (0.00)
Reinforced urn ($r = 1$)	6	+0.29 (0.05)	+0.33 (0.09)	**+0.59** (0.00)	+0.38 (0.09)	+0.54 (0.05)
Reinforced urn ($r = 1$)	50	**+0.61** (0.04)	+0.01 (0.00)	+0.01 (0.00)	+0.02 (0.01)	+0.02 (0.01)

Table 12: Dyck validity, $m = 6$. Rank-out-of-64 of the snapped prompt (1 = enumerated optimum), averaged over $n = 10$ random seeds; standard error in parentheses. Best method per row in bold (within 0.05).

BFT	GCG	PPT-RB (A)	PPT-RB	PPT (A)	PPT
Beta–Bernoulli ($r = 0$)	3.0 (1.33)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)	**1.0** (0.00)
Reinforced urn ($r = 1$)	12.3 (4.65)	21.5 (6.83)	**1.0** (0.00)	17.4 (6.70)	4.3 (3.30)

F.2 PMC AND ESS DIAGNOSTICS

For PPT and PPT-RB, each optimization run logs the importance-weight effective sample size $\mathrm{ESS} := \left(\sum_\ell w_\ell\right)^2 / \sum_\ell w_\ell^2$, with $w_\ell = W(\tilde{k}^{(\ell)};\, \kappa)$, at every step. Low ESS/L indicates the tilted posterior Π_{tilt} has drifted far from Π_0, degrading the IS gradient estimate. Tables 13 and 14 report the final ESS/L (mean, min, max over 10 random seeds) for each (utility, method) configuration at $m = 6$; Tables 15 and 16 report the same at $m = 50$.

Table 13: Final ESS/L (mean / min / max over 10 seeds), Beta–Bernoulli ($r = 0$), $m = 6$.

Utility	PPT-RB (A)			PPT-RB			PPT (A)			PPT		
	mean	min	max	mean	min	max	mean	min	max	mean	min	max
Dyck	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	1.00	0.99	0.96	1.00
$q^\star = 0.0$	0.31	0.31	0.32	0.31	0.31	0.31	0.32	0.31	0.32	0.31	0.31	0.31
$q^\star = 0.1$	0.31	0.31	0.32	0.31	0.31	0.31	0.32	0.31	0.33	0.31	0.31	0.32
$q^\star = 0.2$	0.31	0.31	0.32	0.31	0.31	0.31	0.32	0.31	0.33	0.31	0.31	0.32
$q^\star = 0.3$	0.34	0.32	0.36	0.33	0.32	0.34	0.34	0.32	0.36	0.34	0.32	0.35
$q^\star = 0.4$	0.47	0.46	0.48	0.47	0.46	0.47	0.47	0.46	0.48	0.47	0.46	0.47
$q^\star = 0.5$	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	1.00	1.00	1.00	1.00
$q^\star = 0.6$	0.48	0.47	0.52	0.49	0.48	0.50	0.48	0.46	0.49	0.48	0.48	0.49
$q^\star = 0.7$	0.34	0.32	0.36	0.35	0.33	0.36	0.34	0.32	0.36	0.35	0.33	0.36
$q^\star = 0.8$	0.32	0.31	0.33	0.32	0.32	0.32	0.32	0.32	0.34	0.32	0.32	0.33
$q^\star = 0.9$	0.32	0.31	0.33	0.32	0.32	0.32	0.32	0.31	0.34	0.32	0.32	0.33
$q^\star = 1.0$	0.32	0.31	0.33	0.32	0.32	0.32	0.32	0.31	0.33	0.32	0.32	0.33
$\tau^\star = 0.1$	0.31	0.31	0.32	0.31	0.31	0.31	0.32	0.31	0.32	0.31	0.31	0.31
$\tau^\star = 0.2$	0.31	0.31	0.32	0.31	0.31	0.31	0.32	0.31	0.32	0.31	0.31	0.31
$\tau^\star = 0.3$	0.31	0.31	0.32	0.31	0.31	0.31	0.32	0.31	0.32	0.31	0.31	0.31
$\tau^\star = 0.4$	0.31	0.31	0.32	0.31	0.31	0.31	0.32	0.31	0.32	0.31	0.31	0.31
$\tau^\star = 0.6$	0.32	0.31	0.33	0.32	0.32	0.32	0.32	0.31	0.33	0.32	0.32	0.32
$\tau^\star = 0.7$	0.32	0.31	0.33	0.32	0.32	0.32	0.32	0.31	0.33	0.32	0.32	0.32
$\tau^\star = 0.8$	0.32	0.31	0.33	0.32	0.32	0.32	0.32	0.31	0.33	0.32	0.32	0.32
$\tau^\star = 0.9$	0.32	0.31	0.33	0.32	0.32	0.32	0.32	0.31	0.33	0.32	0.32	0.32

Table 14: Final ESS/L (mean / min / max over 10 seeds), Reinforced urn ($r = 1$), $m = 6$.

Utility	PPT-RB (A)			PPT-RB			PPT (A)			PPT		
	mean	min	max	mean	min	max	mean	min	max	mean	min	max
Dyck	0.22	0.21	0.23	0.20	0.20	0.20	0.22	0.21	0.23	0.22	0.20	0.42
$q^\star = 0.0$	0.31	0.22	0.35	0.41	0.20	0.49	0.31	0.22	0.35	0.42	0.20	0.49
$q^\star = 0.1$	0.31	0.22	0.35	0.41	0.20	0.49	0.32	0.22	0.35	0.42	0.20	0.49
$q^\star = 0.2$	0.31	0.22	0.35	0.41	0.20	0.49	0.32	0.22	0.35	0.42	0.20	0.49
$q^\star = 0.3$	0.26	0.22	0.30	0.25	0.20	0.44	0.27	0.22	0.31	0.25	0.20	0.44
$q^\star = 0.4$	0.22	0.22	0.23	0.20	0.20	0.20	0.23	0.22	0.24	0.20	0.20	0.21
$q^\star = 0.5$	0.22	0.21	0.23	0.20	0.20	0.20	0.22	0.21	0.23	0.20	0.20	0.20
$q^\star = 0.6$	0.24	0.22	0.35	0.24	0.20	0.38	0.24	0.22	0.35	0.24	0.20	0.38
$q^\star = 0.7$	0.28	0.22	0.35	0.25	0.20	0.38	0.29	0.23	0.35	0.25	0.20	0.38
$q^\star = 0.8$	0.32	0.27	0.35	0.32	0.20	0.38	0.32	0.27	0.35	0.32	0.20	0.38
$q^\star = 0.9$	0.32	0.27	0.35	0.34	0.25	0.38	0.32	0.27	0.35	0.34	0.25	0.38
$q^\star = 1.0$	0.32	0.27	0.35	0.34	0.25	0.38	0.32	0.27	0.35	0.34	0.25	0.38
sym-0.1	0.22	0.21	0.23	0.20	0.20	0.20	0.22	0.21	0.23	0.20	0.20	0.20
sym-0.2	0.22	0.21	0.23	0.20	0.20	0.20	0.22	0.21	0.23	0.20	0.20	0.20
sym-0.3	0.22	0.21	0.23	0.20	0.20	0.20	0.22	0.21	0.23	0.20	0.20	0.20
sym-0.4	0.22	0.21	0.23	0.20	0.20	0.20	0.22	0.21	0.23	0.20	0.20	0.20
sym-0.6	0.34	0.28	0.35	0.41	0.38	0.48	0.33	0.28	0.36	0.38	0.26	0.49
sym-0.7	0.34	0.27	0.35	0.41	0.38	0.48	0.33	0.28	0.36	0.38	0.26	0.49
sym-0.8	0.34	0.27	0.35	0.41	0.38	0.48	0.33	0.28	0.36	0.38	0.26	0.49
sym-0.9	0.34	0.27	0.35	0.41	0.38	0.48	0.33	0.28	0.36	0.38	0.26	0.48
sym-1.0	0.34	0.33	0.35	0.44	0.38	0.48	0.34	0.33	0.35	0.44	0.38	0.48
dir-0	0.32	0.27	0.35	0.32	0.22	0.38	0.32	0.27	0.35	0.32	0.21	0.38
dir-1	0.33	0.29	0.35	0.44	0.37	0.49	0.33	0.29	0.35	0.44	0.37	0.49
dir-2	0.32	0.27	0.35	0.34	0.24	0.38	0.32	0.27	0.35	0.34	0.24	0.38
dir-3	0.34	0.27	0.35	0.40	0.25	0.48	0.33	0.27	0.36	0.38	0.25	0.49
dir-4	0.34	0.30	0.35	0.43	0.38	0.48	0.34	0.30	0.35	0.43	0.38	0.49
dir-5	0.22	0.21	0.23	0.20	0.20	0.20	0.22	0.21	0.23	0.20	0.20	0.20
dir-6	0.34	0.31	0.35	0.45	0.38	0.48	0.34	0.31	0.35	0.43	0.38	0.49
dir-7	0.34	0.33	0.35	0.42	0.38	0.48	0.33	0.28	0.36	0.39	0.27	0.48
dir-8	0.25	0.22	0.35	0.24	0.20	0.38	0.25	0.22	0.35	0.24	0.20	0.38
dir-9	0.33	0.29	0.35	0.41	0.20	0.49	0.33	0.29	0.35	0.41	0.20	0.49

Table 15: Final ESS/L (mean / min / max over 10 seeds), Beta–Bernoulli ($r = 0$), $m = 50$.

Utility	PPT-RB (A)			PPT-RB			PPT (A)			PPT		
	mean	min	max	mean	min	max	mean	min	max	mean	min	max
Dyck	1.00	1.00	1.00	1.00	1.00	1.00	0.97	0.84	1.00	0.94	0.86	1.00
$q^\star = 0.0$	0.11	0.10	0.12	0.11	0.11	0.11	0.12	0.10	0.13	0.12	0.11	0.13
$q^\star = 0.1$	0.12	0.10	0.13	0.11	0.11	0.12	0.12	0.10	0.13	0.12	0.11	0.13
$q^\star = 0.2$	0.16	0.13	0.21	0.16	0.12	0.21	0.16	0.13	0.21	0.16	0.12	0.22
$q^\star = 0.3$	0.20	0.13	0.38	0.17	0.13	0.31	0.19	0.13	0.31	0.18	0.13	0.32
$q^\star = 0.4$	0.40	0.21	0.48	0.42	0.24	0.47	0.43	0.23	0.50	0.42	0.23	0.49
$q^\star = 0.5$	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.96	1.00	0.99	0.97	1.00
$q^\star = 0.6$	0.36	0.15	0.49	0.37	0.16	0.48	0.36	0.16	0.50	0.37	0.16	0.51
$q^\star = 0.7$	0.19	0.12	0.33	0.20	0.13	0.33	0.19	0.12	0.32	0.18	0.13	0.33
$q^\star = 0.8$	0.17	0.12	0.22	0.17	0.13	0.21	0.17	0.12	0.23	0.17	0.13	0.21
$q^\star = 0.9$	0.12	0.11	0.13	0.12	0.11	0.13	0.12	0.11	0.14	0.12	0.11	0.13
$q^\star = 1.0$	0.11	0.11	0.12	0.11	0.11	0.12	0.12	0.11	0.14	0.12	0.11	0.13
$\tau^\star = 0.1$	0.11	0.10	0.11	0.11	0.11	0.11	0.12	0.10	0.13	0.12	0.11	0.12
$\tau^\star = 0.2$	0.11	0.10	0.11	0.11	0.11	0.11	0.12	0.10	0.13	0.12	0.11	0.12
$\tau^\star = 0.3$	0.11	0.10	0.11	0.11	0.11	0.11	0.12	0.10	0.13	0.12	0.11	0.12
$\tau^\star = 0.4$	0.11	0.10	0.11	0.11	0.11	0.11	0.12	0.10	0.13	0.12	0.11	0.12
$\tau^\star = 0.6$	0.11	0.11	0.12	0.11	0.11	0.11	0.12	0.11	0.13	0.12	0.11	0.13
$\tau^\star = 0.7$	0.11	0.11	0.12	0.11	0.11	0.11	0.12	0.11	0.13	0.12	0.11	0.13
$\tau^\star = 0.8$	0.11	0.11	0.12	0.11	0.11	0.11	0.12	0.11	0.13	0.12	0.11	0.13
$\tau^\star = 0.9$	0.11	0.11	0.12	0.11	0.11	0.11	0.12	0.11	0.13	0.12	0.11	0.13

Table 16: Final ESS/L (mean / min / max over 10 seeds), Reinforced urn ($r = 1$), $m = 50$.

Utility	PPT-RB (A)			PPT-RB			PPT (A)			PPT		
	mean	min	max	mean	min	max	mean	min	max	mean	min	max
Dyck	0.05	0.04	0.09	0.07	0.06	0.12	0.06	0.04	0.16	0.08	0.06	0.20
$q^\star = 0.0$	0.11	0.09	0.12	0.15	0.11	0.26	0.09	0.03	0.12	0.12	0.04	0.26
$q^\star = 0.1$	0.13	0.09	0.18	0.19	0.12	0.29	0.10	0.03	0.18	0.14	0.04	0.29
$q^\star = 0.2$	0.18	0.03	0.30	0.21	0.03	0.36	0.15	0.03	0.32	0.17	0.04	0.36
$q^\star = 0.3$	0.07	0.02	0.11	0.08	0.03	0.15	0.07	0.03	0.11	0.09	0.04	0.14
$q^\star = 0.4$	0.03	0.02	0.03	0.03	0.03	0.03	0.04	0.03	0.10	0.06	0.04	0.12
$q^\star = 0.5$	0.03	0.02	0.03	0.03	0.03	0.03	0.04	0.03	0.04	0.05	0.04	0.05
$q^\star = 0.6$	0.04	0.02	0.12	0.05	0.03	0.14	0.05	0.03	0.12	0.06	0.04	0.14
$q^\star = 0.7$	0.07	0.03	0.12	0.07	0.04	0.14	0.07	0.03	0.12	0.07	0.04	0.14
$q^\star = 0.8$	0.17	0.03	0.28	0.12	0.04	0.26	0.13	0.03	0.27	0.12	0.04	0.26
$q^\star = 0.9$	0.11	0.03	0.18	0.11	0.03	0.21	0.11	0.03	0.18	0.11	0.03	0.21
$q^\star = 1.0$	0.10	0.03	0.12	0.11	0.03	0.14	0.08	0.03	0.12	0.10	0.03	0.16
sym-0.1	0.03	0.02	0.03	0.03	0.03	0.03	0.03	0.02	0.03	0.03	0.03	0.03
sym-0.2	0.03	0.02	0.03	0.03	0.03	0.03	0.03	0.02	0.03	0.03	0.03	0.03
sym-0.3	0.03	0.02	0.03	0.03	0.03	0.03	0.03	0.02	0.03	0.03	0.03	0.03
sym-0.4	0.03	0.02	0.03	0.05	0.03	0.23	0.03	0.02	0.03	0.03	0.03	0.03
sym-0.6	0.11	0.09	0.12	0.13	0.09	0.15	0.11	0.09	0.12	0.13	0.10	0.15
sym-0.7	0.11	0.10	0.12	0.14	0.12	0.16	0.11	0.09	0.12	0.13	0.09	0.15
sym-0.8	0.11	0.10	0.12	0.14	0.12	0.16	0.11	0.09	0.12	0.13	0.10	0.16
sym-0.9	0.11	0.10	0.12	0.14	0.14	0.16	0.11	0.09	0.12	0.13	0.10	0.16
sym-1.0	0.11	0.11	0.12	0.14	0.14	0.16	0.11	0.10	0.12	0.14	0.11	0.16
dir-0	0.09	0.03	0.12	0.09	0.03	0.14	0.08	0.03	0.12	0.08	0.03	0.14
dir-1	0.10	0.08	0.12	0.13	0.03	0.15	0.09	0.02	0.12	0.12	0.03	0.15
dir-2	0.11	0.09	0.12	0.12	0.08	0.14	0.11	0.09	0.12	0.13	0.10	0.15
dir-3	0.11	0.10	0.12	0.14	0.12	0.16	0.11	0.09	0.12	0.13	0.10	0.15
dir-4	0.11	0.09	0.12	0.14	0.11	0.16	0.11	0.09	0.12	0.14	0.11	0.16
dir-5	0.03	0.02	0.03	0.03	0.03	0.03	0.03	0.02	0.03	0.03	0.03	0.03
dir-6	0.11	0.09	0.12	0.14	0.11	0.16	0.11	0.09	0.12	0.13	0.11	0.16
dir-7	0.11	0.11	0.12	0.14	0.12	0.16	0.11	0.09	0.12	0.14	0.10	0.16
dir-8	0.05	0.02	0.12	0.06	0.03	0.14	0.05	0.02	0.12	0.06	0.03	0.14
dir-9	0.10	0.02	0.12	0.13	0.03	0.15	0.09	0.02	0.12	0.11	0.03	0.15

G Details deferred from the Discussion

Kernel-class misspecification. PPT may also fail under kernel-class misspecification. For example, with κ restricted to the order-0 class for the reinforced urn BFT, prompt tokens are drawn independently across positions, so the optimization can control token frequencies but cannot favor one ordering over another with the same counts, while the posterior over the 1-Markov latent kernel $\tilde{Q}$ updates on transition counts. Performance on targets favoring sequentially structured prompts should therefore degrade substantially. However, the order-0 class can still express constant prompts, so repetition-heavy targets may survive.

Extending to real-scale LLMs. PMC's rollout-and-functional procedure (Appendix C.1) can be attempted on real-scale LLMs by generating unconditional continuations and estimating a functional from each rollout. Interpreting these estimates as latent-prior samples, however, requires convergence to a limiting random measure $\tilde{P}$; the random pmf $\tilde{p}$ and random transition table $\tilde{Q}$ studied here are special cases. We do not expect low-order Markov kernels to capture LLM semantics. Given such a limiting object and a useful measurable functional g, the prior and posterior distributions of $g(\tilde{P})$ are the pushforwards under g of the corresponding laws of $\tilde{P}$. These distributions are mathematically defined even when the parameters indexing $\tilde{P}$ are unknown, although estimating them from finite rollouts is a separate requirement. The persona factors discussed in Section 5 are candidates for such functionals, provided that an appropriate functional can be specified and estimated from rollouts. Applying PPT additionally requires a tractable parametric representation that makes the tilt factor

W and its score with respect to the prompt parameters computable (Section 3). Establishing these requirements for real-scale LLMs remains future work.

H BROADER IMPACTS

Elicitation methods are dual-use: they support safety testing but could inform adversarial prompt search. Our method is restricted to stylized BFTs with binary vocabularies, limiting near-term risk.