BETTER STARTS FOR VALUE-BASED LLM REASONING: REFERENCE-RETURN INITIALIZED TBRM

Anonymous authors
Paper under double-blind review

ABSTRACT

Trajectory Bellman Residual Minimization (TBRM) enables value-based reinforcement learning for large language model reasoning by interpreting token logits as action values. However, initializing the model from a reference policy does not automatically calibrate its value estimate: action-independent shifts leave token probabilities unchanged but alter the log-partition used to represent value. This ambiguity can create a mismatch between the initial value and downstream task returns. We give a unified value-anchoring framework that organizes TBRM and the trajectory residual used by ReVal as different choices of a fixed prompt-wise initial value. Within this framework, we propose *Reference-Return Initialized TBRM* (RI-TBRM), which estimates the reference policy's mean return in an offline annotation pass and reuses the resulting value anchor throughout reinforcement learning. The method applies a fixed offset consistently to action and state values, preserving the initial policy and the trajectory Bellman residual functional without introducing a separate critic. Under reference-policy sampling, exact calibration minimizes the initial residual mean squared error among prompt-wise scalar value initializations and removes the value-mismatch term from the initial expected gradient. We further characterize the estimation error introduced by a finite number of independent calibration rollouts. RI-TBRM achieves the highest six-benchmark average on all three evaluated model backbones.

1 INTRODUCTION

Value-based reinforcement learning offers a way to improve large language model (LLM) reasoning through Bellman consistency, using the autoregressive model to represent both the policy and the value function without an additional critic (Guo et al., 2022; Yuan et al., 2025b). Trajectory Bellman Residual Minimization (TBRM; Yuan et al., 2025b) makes this approach concrete: token logits act as action values, and token-level residuals are summed over a complete response before squaring. Intermediate values cancel, leaving a loss determined by the initial prompt value, response probabilities, and the outcome reward. This provides a value-based counterpart to policy-gradient approaches for learning from verifiable reasoning rewards (Shao et al., 2024; DeepSeek-AI et al., 2025).

A shared policy–value representation raises a specific initialization question. A reference checkpoint provides useful token probabilities, but these probabilities do not determine an absolute value level in downstream reward units. Adding a common scalar to all action logits preserves their softmax probabilities while shifting their log-partition. Consequently, *the same initial policy can yield different Bellman residuals and different learning updates*. The relevant design choice is not only which policy to start from, but also where to anchor its initial value.

Reference-shifted formulations already address part of this problem (Clavier et al., 2025; Wang et al., 2026). ReVal explicitly identifies TBRM's initialization miscalibration and subtracts the reference log-partition, yielding a zero initial residual when rewards are identically zero. Zero anchoring removes the inherited logit level. A reference policy's existing task performance provides a further source of information: its expected return quantifies task performance separately for each prompt.

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.

This motivates choosing an initial value that reflects the starting policy's prompt-dependent task performance.

We study this question through a *unified value-anchoring framework*. A fixed scalar $v(x)$ specifies the initial value for each prompt while a shared translation of action and state values preserves the policy. Choosing the inherited reference log-partition recovers TBRM, while choosing zero yields the trajectory residual used by ReVal. The framework isolates one design variable—the initial value anchor—and makes its effect on residual learning explicit.

Our analysis gives a criterion for choosing that anchor. At exact reference-weight initialization and under reference-policy sampling, the expected squared residual decomposes into return variance and squared value mismatch. More importantly, the expected full-residual gradient contains an additional term proportional to that mismatch, through the trainable prompt log-partition. Setting $v(x)$ to the reference policy's expected scaled return removes this term. Thus the reference-return baseline has a role beyond ordinary policy-gradient centering: it changes the initial expected gradient through the value path. The resulting criterion connects a measurable property of the reference policy to the initial direction of Bellman residual learning.

We implement this criterion as *Reference-Return Initialized TBRM* (RI-TBRM). Because the reference policy, prompts, and reward function are fixed, their prompt-wise return statistics form a reusable annotation of the training set. We generate these annotations offline from independent reference responses, cache the mean reward together with the reference log-partition, and read the cache throughout policy optimization. Annotation samples are independent across prompts and responses, so this stage can be sharded across inference workers and prepared independently of the RL loop. The same cache can be reused by runs that share the reference checkpoint, dataset, reward, and sampling convention. We quantify its finite-sample estimation error and show invariance to prompt-wise reward translations.

We evaluate RI-TBRM against TBRM, ReVal, and GRPO. On six reasoning benchmarks, RI-TBRM achieves the highest average on each backbone. Our contributions are:

- A unified value-anchoring framework that recovers TBRM and the trajectory residual used by ReVal as different initial value choices.
- An initialization analysis linking the anchor to residual risk and the expected full-residual gradient, which yields a prompt-conditional reference-return calibration criterion.
- RI-TBRM, a cached reference-return implementation of this criterion, with finite-sample analysis and evaluations of reasoning performance.

2 Related Work

Policy optimization for language models. PPO optimizes a surrogate policy objective using sampled trajectories (Schulman et al., 2017b). For language-model post-training, GRPO replaces a learned critic with group-relative reward estimates, while REINFORCE leave-one-out (RLOO) uses the other sampled responses to construct a baseline (Shao et al., 2024; Ahmadian et al., 2024). These methods center policy-gradient advantages. In our setting, the baseline calibrates a value term inside a squared Bellman residual and influences the expected gradient through the trainable log-partition. Our analysis characterizes this value-path contribution and the corresponding initial residual risk.

Value-based learning and trajectory consistency. Entropy-regularized RL connects soft value functions with stochastic policies (Schulman et al., 2017a). Path Consistency Learning uses multi-step consistency relations to learn from both on- and off-policy trajectories (Nachum et al., 2017). Guo et al. extend soft Q-learning to text generation with sparse rewards (Guo et al., 2022). TBRM subsequently formulates language-model reasoning as trajectory Bellman residual minimization using the model's logits as Q-values (Yuan et al., 2025b). Our framework exposes the initial value level within this shared parameterization, connecting trajectory consistency to an explicit choice of value anchor.

Reference-shifted Bellman objectives. ShiQ derives shifted Bellman equations for language-model fine-tuning, and its sequence-level variant contains the difference between current and ref-

erence log-partitions (Clavier et al., 2025). ReVal uses this difference in its trajectory residual and couples it with its replay-based training procedure (Wang et al., 2026). In our anchor family, $v(x) = 0$ yields the ReVal trajectory residual and $v(x) = U_{\mathrm{ref}}(x)$ yields TBRM. RI-TBRM chooses $v(x) = \widehat{r}_{\mathrm{ref}}(x)/\beta$, which is equivalently a prompt-dependent centering of the reference-subtracted reward. This view organizes the three residual constructions by their initial value anchor.

Value initialization and target normalization. VC-PPO addresses critic initialization through value pretraining on responses from a fixed supervised-fine-tuned policy, together with a decoupled advantage-estimation procedure (Yuan et al., 2025a). It shares the motivation that the initial policy and value estimates should be compatible, but operates through a separately optimized critic. Our calibration instead shifts a logit-derived value without fitting an additional value network. PopArt adaptively normalizes value targets while preserving unnormalized predictions (van Hasselt et al., 2016). In contrast, our baseline is a fixed, prompt-dependent translation tied to reference-policy returns, rather than an adaptive normalization of target scale.

3 PRELIMINARIES

A prompt $x \sim \rho$ initializes $s_1 = x$. Appending token a_h gives the deterministic transition $s_{h+1} = (x, a_{1:h})$. A complete response $y = a_{1:H_y}$, with $H_y \le H$, has reward $R(x, y) = \sum_{h=1}^{H_y} r(s_h, a_h)$ and probability $\pi(y \mid x) = \prod_{h=1}^{H_y} \pi(a_h \mid s_h)$ under a specified stopping convention. Rewards may be terminal-only, and terminal values are zero. For a fixed reference policy π_{ref} and coefficient $\beta > 0$, the objective is

$$J_\beta(\pi) = \mathbb{E}_{x\sim\rho,\, y\sim\pi(\cdot|x)} \left[R(x, y) - \beta \log \frac{\pi(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)} \right]. \tag{1}$$

We assume a common response support with positive reference probability and write $Y(x, y) = R(x, y)/\beta$ and $D_\theta(x, y) = \log[\pi_\theta(y \mid x)/\pi_{\mathrm{ref}}(y \mid x)]$. The conditional scaled policy value is $V_\beta^\pi(x) = \mathbb{E}_\pi[Y - \log(\pi/\pi_{\mathrm{ref}}) \mid x]$.

Logit-derived values. TBRM uses token logits $z_\theta(s, a)$ as action values. Their log-partition and induced policy are

$$U_\theta(s) = \log \sum_a e^{z_\theta(s,a)}, \qquad \log \pi_\theta(a \mid s) = z_\theta(s, a) - U_\theta(s). \tag{2}$$

We use U_θ for the model's logit-derived value and $V_\beta^{\pi_\theta}$ for its expected KL-regularized return. We use unit-temperature logits, with other fixed temperatures incorporated consistently into both logits and probabilities.

Trajectory Bellman residuals. Dividing Eq. (1) by β gives transformed token reward $r_\beta(s, a) = r(s, a)/\beta + \log \pi_{\mathrm{ref}}(a \mid s)$. For action values Q and their induced soft value $V_Q(s) = \log \sum_a e^{Q(s,a)}$, the soft Bellman residual is

$$\delta_Q(s, a, s') = Q(s, a) - \frac{r(s, a)}{\beta} - \log \pi_{\mathrm{ref}}(a \mid s) - V_Q(s'). \tag{3}$$

TBRM sums these residuals over a complete response before squaring (Yuan et al., 2025b):

$$\mathcal{L}_{\mathcal{D}}(Q) = \frac{1}{2|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \left[\sum_{h=1}^{H_y} \delta_Q(s_h, a_h, s_{h+1}) \right]^2. \tag{4}$$

This differs from summing individually squared token residuals. The factor $1/2$ only rescales the original TBRM objective. Intermediate soft values cancel, leaving the prompt value, response log-ratio, and total reward; Appendix B.2 gives the full derivation.

4 A UNIFIED VALUE-ANCHORING FRAMEWORK

We expose the initial value as an explicit design variable. Let $v(x)$ be a prompt-wise scalar fixed before RL and held constant during optimization. The trainable weights start at $\theta_0 = \theta_{\mathrm{ref}}$, with identical forward-pass and probability conventions; the reference model remains frozen.

4.1 POLICY-PRESERVING VALUE ANCHORING

For every nonterminal prefix s of prompt x, define

$$\begin{aligned} Q_{\theta,v}(s,a) &= z_\theta(s,a) - U_{\mathrm{ref}}(x) + v(x), \\ V_{\theta,v}(s) &= U_\theta(s) - U_{\mathrm{ref}}(x) + v(x). \end{aligned} \tag{5}$$

The offset is shared across actions and nonterminal prefixes, while $V_{\theta,v}(s_{H_y+1}) = 0$ is defined separately. Translation of log-sum-exp gives

$$\log \sum_a e^{Q_{\theta,v}(s,a)} = V_{\theta,v}(s), \qquad Q_{\theta,v}(s,a) - V_{\theta,v}(s) = \log \pi_\theta(a \mid s). \tag{6}$$

Thus every anchor induces the same policy at fixed weights. At the initial prompt,

$$V_{\theta_0,v}(x) = v(x), \qquad Q_{\theta_0,v}(x,a) = \log \pi_{\mathrm{ref}}(a \mid x) + v(x). \tag{7}$$

The anchor specifies the initial prompt value while preserving the reference model's token preferences. Later prefixes share this prompt-level offset, and their values evolve through the model's trainable logits.

4.2 TBRM AND THE REVAL RESIDUAL AS ANCHOR CHOICES

Substituting Eq. (5) into the trajectory functional gives

$$\Delta_{\theta,v}(x,y) = U_\theta(x) - U_{\mathrm{ref}}(x) + v(x) + D_\theta(x,y) - Y(x,y), \tag{8}$$

$$\mathcal{L}_v(\theta;\mathcal{D}) = \frac{1}{2|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \Delta_{\theta,v}(x,y)^2. \tag{9}$$

Only the starting value remains after trajectory cancellation. Table 1 identifies the three choices studied here: TBRM inherits the reference log-partition, the trajectory residual used by ReVal anchors at zero, and RI-TBRM anchors at the estimated scaled reference return $\widehat{b}_K$ defined in Section 6.

Table 1: A common trajectory residual family with different fixed value anchors.

Trajectory objective	Anchor $v(x)$	Initial prompt value
TBRM (Yuan et al., 2025b)	$U_{\mathrm{ref}}(x)$	Inherited log-partition
ReVal trajectory residual (Wang et al., 2026)	0	Zero
RI-TBRM	$\widehat{b}_K(x)$	Estimated reference return/β

In particular, $v = U_{\mathrm{ref}}$ gives $\Delta = U_\theta + D_\theta - Y$, whereas $v = 0$ gives $\Delta = U_\theta - U_{\mathrm{ref}} + D_\theta - Y$. A nonzero anchor can equivalently be viewed as a fixed reward translation $R \mapsto R - \beta v$ in the latter objective. The shared parameterization preserves the induced policy at fixed weights, while the anchor changes the residual and its gradient.

5 CHOOSING THE INITIAL VALUE ANCHOR

The framework separates policy initialization from value anchoring. We now ask which anchor matches the starting policy and how mismatch affects the initial update. Condition on a prompt x and define

$$b(x) = \mathbb{E}_{\mathrm{ref}}[Y \mid x] = V_\beta^{\pi_{\mathrm{ref}}}(x), \qquad \sigma^2(x) = \mathrm{Var}_{\mathrm{ref}}(Y \mid x). \tag{10}$$

The equality for b follows because the reference policy's KL term is zero. We analyze the fixed-reference sampling loss

$$\mathcal{L}_v^{\mathrm{ref}}(\theta \mid x) = \tfrac{1}{2}\mathbb{E}_{y\sim\pi_{\mathrm{ref}}(\cdot|x)}[\Delta_{\theta,v}(x,y)^2]. \tag{11}$$

All results assume exact reference-weight initialization, fixed reference quantities, and full residual differentiation with sampled responses held fixed. Appendix B.1 states the regularity assumptions and provides the proofs.

5.1 A REFERENCE-RETURN CRITERION FOR RESIDUAL CALIBRATION

Proposition 1 (Initial residual calibration)**.** *If $\sigma^2(x) < \infty$, every fixed scalar anchor $v(x)$ satisfies*

$$\begin{aligned}\mathbb{E}_{\mathrm{ref}}[\Delta_{\theta_0,v} \mid x] &= v(x) - b(x),\\ \mathbb{E}_{\mathrm{ref}}[\Delta^2_{\theta_0,v} \mid x] &= \sigma^2(x) + (v(x) - b(x))^2.\end{aligned} \tag{12}$$

The exact reference mean $v(x) = b(x)$ uniquely minimizes the initial residual MSE among prompt-wise scalar value anchors and makes its conditional mean zero.

At initialization, Eq. (8) reduces to $v - Y$, giving the variance–bias decomposition above. TBRM and zero anchoring have risks $\sigma^2 + (U_{\mathrm{ref}} - b)^2$ and $\sigma^2 + b^2$, respectively. Exact calibration removes the squared mismatch and leaves the irreducible return variance $\sigma^2(x)$. A single dataset-wide anchor also retains between-prompt variation $\mathrm{Var}_x\, b(x)$ even at its optimum (Appendix B.3), motivating prompt-conditional calibration.

5.2 WHY THE ANCHOR CHANGES THE INITIAL EXPECTED GRADIENT

Let $g(x) = \nabla_\theta U_\theta(x)\big|_{\theta_0}$ and $q(x) = \nabla_\theta \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}[Y(x,y)]\big|_{\theta_0}$. The latter is the initial expected scaled-reward gradient, whose sampling distribution varies with the policy.

Proposition 2 (Initial full-residual gradient)**.** *Under the assumptions above,*

$$\nabla_\theta \mathcal{L}_v^{\mathrm{ref}}(\theta \mid x)\big|_{\theta_0} = \underbrace{(v(x) - b(x))g(x)}_{\textit{value-mismatch term}} - \underbrace{q(x)}_{\textit{scaled-reward gradient}}. \tag{13}$$

Exact calibration yields $\nabla_\theta \mathcal{L}_b^{\mathrm{ref}}(\theta_0 \mid x) = -q(x) = -\beta^{-1}\, \nabla_\theta J_\beta(\pi_\theta \mid x)\big|_{\theta_0}$.

To see the mechanism, write $S = \nabla_\theta \log \pi_\theta(y \mid x)\big|_{\theta_0}$. The per-response gradient is $(v - Y)(g + S)$. Since $\mathbb{E}_{\mathrm{ref}} S = 0$, the baseline's score-function contribution vanishes in expectation, but its contribution through the trainable log-partition g does not. Reference-return anchoring removes this additional component while retaining the trainable value path. Proposition 2 characterizes the expected loss gradient at θ_0; Section 7 reports reasoning performance and training dynamics in terms of rewards, the learned value correction, and response truncation.

6 REFERENCE-RETURN INITIALIZED TBRM

The default RI-TBRM implementation uses an independently estimated, frozen anchor, separating offline annotation from policy optimization. Section 7.4 also examines a refreshed-reference, rollout-reuse variant.

6.1 OFFLINE REFERENCE-RETURN ANNOTATION

For each training prompt, generate K independent responses from the frozen reference policy and compute

$$\widehat{r}_{\mathrm{ref}}(x) = \frac{1}{K}\sum_{k=1}^{K} R(x, y^{(k)}), \quad y^{(k)} \overset{\mathrm{iid}}{\sim} \pi_{\mathrm{ref}}(\cdot \mid x), \quad \widehat{b}_K(x) = \widehat{r}_{\mathrm{ref}}(x)/\beta. \tag{14}$$

For binary rewards, $\widehat{r}_{\mathrm{ref}}$ is the fraction of correct reference responses. The annotation pass produces two cached scalars per training prompt: $\widehat{r}_{\mathrm{ref}}(x)$ and $U_{\mathrm{ref}}(x)$. Because the reference checkpoint and

Algorithm 1 Reference-Return Initialized TBRM (RI-TBRM)

Require: Training prompts $\mathcal{X}$, reference weights θ_{ref}, reward R, coefficient β, reference sample count K

Freeze θ_{ref}
Offline reference annotation
for $x \in \mathcal{X}$ **do**
  Sample K independent reference responses; compute $\widehat{r}_{\mathrm{ref}}(x)$ via Eq. (14)
  Cache $\widehat{r}_{\mathrm{ref}}(x)$ and $U_{\mathrm{ref}}(x)$
end for
Freeze the cache and discard the calibration responses
Policy optimization with the cached annotations
Initialize $\theta \leftarrow \theta_{\mathrm{ref}}$
for each training iteration **do**
  Generate fresh responses with π_θ to construct $\mathcal{D}_t$
  Evaluate rewards and reference response log-probabilities
  for each scheduled update on $\mathcal{D}_t$ **do**
    Read the cached scalars; recompute $U_\theta(x)$ and $\log \pi_\theta(y \mid x)$
    Update θ using the full gradient of Eq. (16)
  end for
end for
return π_θ

task specification are fixed, these values form a static annotation of the training set. Prompts and reference responses are independent units of inference, allowing the annotation pass to be sharded across workers and prepared independently of policy optimization. Compatible training runs read the same cache throughout optimization.

Set $v = \widehat{b}_K$ in Eq. (5) and write the resulting values as $\widetilde{Q}_\theta$ and $\widetilde{V}_\theta$. They satisfy

$$\widetilde{V}_{\theta_0}(x) = \widehat{b}_K(x), \qquad \widetilde{Q}_{\theta_0}(x,a) = \log \pi_{\mathrm{ref}}(a \mid x) + \widehat{b}_K(x). \tag{15}$$

6.2 Policy optimization with cached anchors

The training objective is

$$\mathcal{L}^{\mathrm{RI}}_{\mathcal{D}}(\theta) = \frac{1}{2|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \left[U_\theta(x) - U_{\mathrm{ref}}(x) + D_\theta(x,y) + \frac{\widehat{r}_{\mathrm{ref}}(x) - R(x,y)}{\beta} \right]^2. \tag{16}$$

The reference model and annotations stay fixed, while gradients pass through both U_θ and the response log-probabilities. Each update reads the cached anchor alongside the usual policy rollouts, reward evaluation, reference scoring, and backpropagation. The model learns a value correction relative to the same reference return throughout optimization.

Reusable offline cache. Reference-return estimation is a dataset annotation step for a fixed reference policy and reward function. Once generated, the cache is reused at every RL iteration and can be shared across runs that use the same reference checkpoint, prompts, formatting, reward, and sampling convention. Only the cached scalars are consumed during optimization; the reference responses themselves can be discarded after annotation. Storing the raw mean reward also lets the training code apply the chosen β when constructing the anchor.

6.3 Finite-sample annotation accuracy

Corollary 3 (Independent finite-rollout calibration)**.** *With $K \geq 1$ independent reference calibration returns and an independent update response,*

$$\mathbb{E}_{\mathrm{cal}}[(\widehat{b}_K - b)^2 \mid x] = \frac{\sigma^2(x)}{K}, \qquad \mathbb{E}_{\mathrm{cal,ref}}[\Delta^2_{\theta_0,\widehat{b}_K} \mid x] = \left(1 + \frac{1}{K}\right)\sigma^2(x). \tag{17}$$

For $G_{\widehat{b}_K} = \nabla_\theta \mathcal{L}^{\mathrm{ref}}_{\widehat{b}_K}(\theta_0 \mid x)$, $\mathbb{E}_{\mathrm{cal}} G_{\widehat{b}_K} = -q$ and $\mathbb{E}_{\mathrm{cal}}\|G_{\widehat{b}_K} + q\|^2 = (\sigma^2/K)\|g\|^2$.

The reference sample count directly controls the estimation component of initial risk: increasing K reduces this component as $1/K$. Relative to a deterministic anchor $v(x)$, the expected initial MSE improves exactly when $\sigma^2(x)/K < (v(x) - b(x))^2$. For binary rewards with reference success probability $p(x) > 0$, improvement over zero anchoring occurs when $p(x) > 1/(K+1)$. This criterion connects reference-policy success, sample size, and initial residual risk.

For a realized cache, the initial conditional mean residual is $\widehat{b}_K - b$. The gradient expression quantifies variation of the conditional expected gradient across calibration sets. Independent annotation and update responses yield the moments in Corollary 3; Appendix B.5 derives the corresponding self-inclusion terms for shared samples.

6.4 Reward translations and target consistency

If every response to prompt x receives an additional constant $k(x)$, the same calibration responses give $\widehat{r}'_{\mathrm{ref}}(x) = \widehat{r}_{\mathrm{ref}}(x) + k(x)$. Hence

$$R'(x,y) - \widehat{r}'_{\mathrm{ref}}(x) = R(x,y) - \widehat{r}_{\mathrm{ref}}(x), \tag{18}$$

and the RI-TBRM loss is unchanged for every θ, even at finite K. A constant-reward prompt has zero initial residual-loss gradient. More generally, every fixed anchor yields the same KL-regularized optimal policy at a full-support zero-residual solution (Appendix B.6). Value anchoring therefore changes the starting point and residual geometry while preserving this consistency target.

7 Experiments

We compare RI-TBRM with TBRM, ReVal, and GRPO on reasoning benchmarks, and examine rewards, the learned value correction, and response truncation during training.

7.1 Experimental setup

Models, data, and comparisons. We evaluate Qwen2.5-Math-7B (Yang et al., 2024), Qwen3-4B, and Qwen3-8B (Yang et al., 2025), trained on DAPO-Math-17K (Yu et al., 2025) with binary final-answer rewards. Each backbone uses the same initial checkpoint across methods. TBRM and RI-TBRM use the value-based implementation described in this paper. ReVal is run with the authors' complete official implementation, including its method-specific training procedure. Across methods we align the model checkpoint, training data, reward function, rollout limits, and evaluation protocol. We also evaluate the Base checkpoints and GRPO (Shao et al., 2024), with the available comparisons listed in Table 2.

Reference annotation and evaluation. The main RI-TBRM runs use $K = 8, 4, 2$ independent reference responses per training prompt for the three backbones, respectively. The resulting offline annotations are cached and reused throughout optimization. Following TBRM, we report Avg@32 on AIME24, AIME25, and AMC23, and greedy Avg@1 on MATH500, Minerva Math, and OlympiadBench. Avg is their arithmetic mean. Appendix A specifies the training configurations, seed, evaluation protocol, and reference-annotation settings.

7.2 Main Results

Table 2 shows that RI-TBRM obtains the highest six-benchmark average on each backbone. Its average scores are 31.47, 59.00, and 61.04 on Qwen2.5-Math-7B, Qwen3-4B, and Qwen3-8B. Relative to TBRM, these are gains of 0.77, 0.82, and 1.59 points; relative to ReVal, the gains are 0.96, 1.05, and 0.38 points. On the two backbones with GRPO results, the gains over GRPO are 0.37 and 0.84 points.

The aggregate gains span different model scales and initial reasoning capabilities. RI-TBRM achieves the best or tied-best AIME24 score on all three backbones, and the highest MATH500 score on both Qwen3 models. ReVal obtains the highest Minerva Math score on Qwen3-8B. These task-level results accompany the consistent ordering of the aggregate score in favor of reference-return anchoring.

Table 2: Performance of RI-TBRM and baselines on Qwen2.5-Math-7B, Qwen3-4B, and Qwen3-8B. AIME24, AIME25, and AMC23 are evaluated with Avg@32, while MATH500, Minerva Math, and OlympiadBench are evaluated with Avg@1. Avg denotes the arithmetic mean over the six benchmarks.

Model	AIME24	AIME25	AMC23	MATH500	Minerva	Olympiad	Avg
Qwen2.5-Math-7B	9.90	2.60	31.17	48.80	13.60	19.70	20.96
GRPO	18.65	6.56	**54.45**	59.00	18.01	**29.93**	31.10
TBRM	19.79	7.29	52.66	59.00	16.91	28.59	30.70
ReVal	19.58	**8.23**	52.81	57.60	17.28	27.56	30.51
RI-TBRM	**20.83**	7.92	53.52	**59.60**	**18.38**	28.59	**31.47**
Qwen3-4B	60.10	53.02	85.94	71.20	30.51	46.52	57.88
GRPO	60.42	52.92	87.73	70.80	**30.88**	46.22	58.16
TBRM	60.63	53.33	**88.20**	69.80	29.41	47.70	58.18
ReVal	61.77	**55.21**	86.02	69.00	28.31	47.41	57.95
RI-TBRM	**62.81**	**55.21**	87.66	**71.40**	29.04	**47.85**	**59.00**
Qwen3-8B	64.06	54.27	88.98	70.40	31.62	43.70	58.84
TBRM	64.48	54.17	88.59	73.40	31.62	44.44	59.45
ReVal	64.79	56.77	89.77	73.60	**33.09**	45.93	60.66
RI-TBRM	**65.10**	**57.92**	**90.16**	**74.40**	31.99	**46.67**	**61.04**

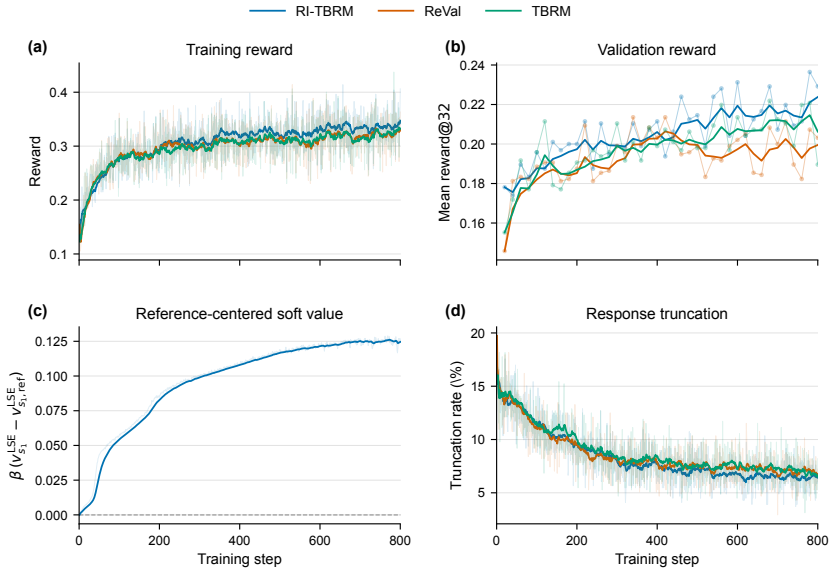


Figure 1: Training dynamics on Qwen2.5-Math-7B. **(a)** Training reward. **(b)** Validation mean@32 reward. **(c)** The learned reference-centered log-partition correction $\beta(U_\theta(x) - U_{\mathrm{ref}}(x))$ for RI-TBRM, labeled with the original v^{LSE} notation in the plot. **(d)** Response truncation rate. Dark curves are smoothed; lightly shaded curves and markers show the original measurements.

7.3 Training dynamics

Figure 1 compares the Qwen2.5-Math-7B training trajectories. We use a 25-step exponential moving average for training metrics and a five-point moving average for validation reward, retaining the original measurements. RI-TBRM's average training reward rises from 0.232 in the first 10% of training to 0.338 in the final 10%, compared with final values of 0.325 for ReVal and 0.326 for TBRM. Its best validation mean@32 reward is 0.236, compared with 0.214 and 0.222, respectively.

The learned reference-centered log-partition correction increases from approximately 0.028 to 0.125, showing the evolution of the trainable value component around the fixed reference-return anchor. Over the same run, the truncation rate decreases from 13.1% to 6.5%. The training trajec-

tory therefore combines higher rewards, a growing learned value correction, and fewer responses reaching the length limit.

7.4 Efficiency Analysis

Reference-return anchoring admits two reference schedules: a frozen reference with independent offline calibration, or a refreshed reference calibrated from training rollouts. Both anchor the value to prompt-wise reference returns, but only the former uses a separate calibration sample.

Offline calibration. The default variant estimates each prompt's reference return once from K independent responses of a frozen policy. The reference returns and log-partitions then remain fixed throughout optimization. These statistics can be reused across runs sharing the same reference policy, prompts, reward, and sampling convention. Calibration is therefore a one-time sampling cost rather than an additional sampling stage at every update; K does not change the per-update policy-optimization complexity.

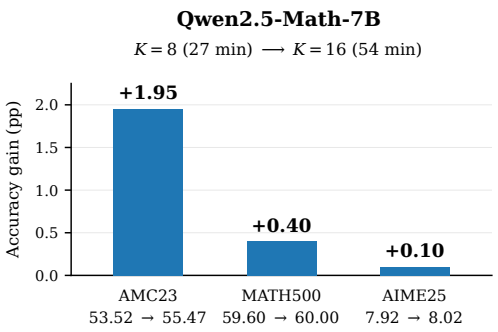


Figure 2: Offline-cache variant on Qwen2.5-Math-7B. Increasing K from 8 to 16 raises one-time pre-sampling from 27 to 54 minutes. Bars show gains in percentage points; labels give $K = 8 \rightarrow K = 16$ scores. AMC23 and AIME25 use Avg@32; MATH500 uses greedy Avg@1. Times exclude RL training.

On Qwen2.5-Math-7B, pre-sampling takes 27 minutes for $K = 8$ and 54 minutes for $K = 16$. The additional 27 minutes accompanies gains of 1.95, 0.40, and 0.10 points on AMC23, MATH500, and AIME25, respectively (Figure 2). Under iid sampling, anchor-estimation MSE is $\sigma^2(x)/K$ (Appendix B.5), so doubling K halves this error; it does not guarantee downstream gains.

Rollout reuse without separate pre-sampling. At iteration t, the current policy π_t generates training responses and serves as the local reference. For each prompt, their mean reward estimates the reference return. Holding this estimate and π_t fixed, we minimize the trajectory Bellman residual on the same responses; the updated policy becomes the next iteration's reference. Thus, one batch supplies both calibration and optimization data, without additional reference-response generation or reward evaluation. On Qwen2.5-Math-7B, rollout reuse attains a six-benchmark average of 31.74, versus 31.47 for the frozen $K = 8$ cache. The recorded training-run elapsed times are 4 h 34 min and 8 h 00 min, respectively: rollout reuse takes 42.9% less time in these runs. Appendix C provides the algorithm, results, and shared-sample analysis. These measurements support competitive accuracy with a shorter observed training run; they do not isolate the effect of removing pre-sampling from other training-configuration changes. Independent pre-sampling is therefore an optional calibration strategy, not an intrinsic requirement of reference-return anchoring.

8 Conclusion

We presented a value-anchoring framework in which TBRM and the ReVal trajectory residual correspond to inherited-value and zero-value initialization. Exact reference-return anchoring minimizes the initial residual MSE among prompt-wise scalar anchors and removes the value-mismatch term from the initial expected gradient. RI-TBRM implements this criterion through an offline reference-return annotation cache that is reused throughout policy optimization and across compatible runs. Finite-sample analysis quantifies annotation accuracy, while experiments show the highest six-benchmark average in the main comparison on all three backbones. A refreshed-reference variant also achieves competitive accuracy without separate pre-sampling. Reference-return anchoring thus supports both reusable offline calibration and rollout reuse.

AI USE STATEMENT

Generative AI tools assisted with language polishing and manuscript restructuring. The authors are responsible for verifying the mathematical statements, citations, experimental descriptions, and final text.

REPRODUCIBILITY STATEMENT

Algorithm 1 describes the annotation and training procedure. Appendix A reports the optimization configuration, rollout settings, evaluation protocol, and reference-annotation settings. Appendix B provides the complete-trajectory derivation, assumptions, and proofs supporting the initialization analysis. Appendix C presents the refreshed-reference algorithm, its full evaluation results, and its distinction from independent offline calibration.

REFERENCES

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.662. URL https://aclanthology.org/2024.acl-long.662/.

Pierre Clavier, Nathan Grinsztajn, Raphaël Avalos, Yannis Flet-Berliac, Irem Ergun, Omar D. Domingues, Eugene Tarassov, Olivier Pietquin, Pierre H. Richemond, Florian Strub, and Matthieu Geist. ShiQ: Bringing back bellman to LLMs, 2025. URL https://arxiv.org/abs/2505.11081.

DeepSeek-AI et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Han Guo, Bowen Tan, Zhengzhong Liu, Eric Xing, and Zhiting Hu. Efficient (soft) Q-learning for text generation with limited good data. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 6969–6991. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.findings-emnlp.518. URL https://aclanthology.org/2022.findings-emnlp.518/.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *arXiv preprint arXiv:2402.14008*, 2024.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. *arXiv preprint arXiv:2206.14858*, 2022.

Ofir Nachum, Mohammad Norouzi, Kelvin Xu, and Dale Schuurmans. Bridging the gap between value and policy based reinforcement learning, 2017. URL https://arxiv.org/abs/1702.08892.

John Schulman, Xi Chen, and Pieter Abbeel. Equivalence between policy gradients and soft Q-learning, 2017a. URL https://arxiv.org/abs/1704.06440.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017b. URL https://arxiv.org/abs/1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Hado van Hasselt, Arthur Guez, Matteo Hessel, Volodymyr Mnih, and David Silver. Learning values across many orders of magnitude, 2016. URL https://arxiv.org/abs/1602.07714.

Peng-Yuan Wang, Ziniu Li, Tian Xu, Bohan Yang, Tian-Shuo Liu, ChenYang Wang, Xiong-Hui Chen, Yi-Chen Li, Tianyun Yang, Congliang Chen, and Yang Yu. Off-policy value-based reinforcement learning for large language models, 2026. URL https://arxiv.org/abs/2603.23355.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/a4277440d50f1f15d2cb4c14f7e0c0d2-Abstract-Conference.html.

Yufeng Yuan, Yu Yue, Ruofei Zhu, Tiantian Fan, and Lin Yan. What's behind PPO's collapse in long-CoT? value optimization holds the secret, 2025a. URL https://arxiv.org/abs/2503.01491.

Yurun Yuan, Fan Chen, Zeyu Jia, Alexander Rakhlin, and Tengyang Xie. Trajectory bellman residual minimization: A simple value-based method for LLM reasoning. In *Advances in Neural Information Processing Systems*, volume 38, 2025b. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/a9a3c5df70368a2f8c3fd2ed12620203-Abstract-Conference.html.

Table 3: Optimization and rollout settings used in the experiments.

Hyperparameter	Value
Optimizer	AdamW
Adam coefficients	$(\beta_1, \beta_2) = (0.9, 0.999)$
Weight decay	0.01
Learning-rate schedule	Constant
Warm-up steps	0
Maximum gradient norm	1.0
Policy optimization epochs	1
Training precision	`bfloat16`
Rollout temperature	1.0
Rollout top-p	1.0
Entropy coefficient	0
Reward/KL coefficient β	0.002
Reward	Binary rule-based verifier
Random seed	42
Training framework	VERL with FSDP
Rollout backend	vLLM
Hardware	8× NVIDIA H100 80GB

A Additional Experimental Details

The configurations below describe the main fixed-reference comparisons. The supplemental refreshed-reference workflow is reported separately in Appendix C.

Training data and reward. We train all models on DAPO-Math-17K, which contains 17,917 unique mathematical problems. Each problem is formatted with the following instruction: "Solve the following math problem step by step. The last line of your response should be of the form `Answer: $Answer`." The model-specific chat template provided by the corresponding tokenizer is then applied. We employ a rule-based verifier as the reward function, assigning a reward of 1 to a correct final answer and 0 otherwise. We do not apply an additional response-length penalty.

Optimization. Our implementation is based on the VERL framework and uses vLLM for rollout generation. All models are fully fine-tuned with FSDP on eight NVIDIA H100 80GB GPUs using `bfloat16` precision and gradient checkpointing. We use AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.999$, weight decay 0.01, and a maximum gradient norm of 1.0. We use a constant learning-rate schedule without warm-up. Each rollout batch is optimized for one epoch. The prompt batch and PPO mini-batch have the same size, corresponding to one full-batch update per training iteration. Dynamic token-based batching is enabled during policy updates.

During training, responses are sampled with temperature 1.0 and top-$p = 1.0$. We set the entropy coefficient to zero. RI-TBRM and TBRM use the critic-free value-based implementation described in this paper with reward/KL scaling coefficient $\beta = 0.002$. ReVal uses the authors' complete official implementation, including its method-specific training components, with the model, data, reward, rollout limits, and evaluation protocol aligned to the corresponding backbone. GRPO uses the standard clipped surrogate objective with a clipping ratio of 0.2 and a low-variance KL loss with coefficient 0.002.

Backbone-specific configurations. Table 4 summarizes the backbone-specific configurations. Here, B denotes the prompt batch size, B_{mini} is the PPO mini-batch size, B_{micro} is the fallback per-GPU micro-batch size, n is the number of responses sampled per prompt, and K_{ref} is the number of samples used to estimate reference-policy statistics when required. L_p and L_r denote the maximum prompt and response lengths, respectively.

Value-based baselines. RI-TBRM and TBRM are trained with the implementation described in this paper. The ReVal baseline uses the authors' complete official implementation and retains its method-specific training procedure. Section 4 provides an objective-level view in which the ReVal trajectory residual corresponds to $v = 0$, TBRM to $v = U_{\text{ref}}$, and RI-TBRM to $v = \widehat{b}_K$. Across

Table 4: Backbone-specific training hyperparameters.

Backbone	Method	LR	B	B_{mini}	B_{micro}/GPU	n	K_{ref}	(L_p, L_r)	Steps
Qwen2.5-Math-7B	RI-TBRM	2×10^{-6}	128	128	8	4	8	$(1024, 2048)$	800
Qwen2.5-Math-7B	ReVal	2×10^{-6}	128	128	8	4	–	$(1024, 2048)$	800
Qwen2.5-Math-7B	TBRM	2×10^{-6}	128	128	8	4	–	$(1024, 2048)$	800
Qwen2.5-Math-7B	GRPO	2×10^{-6}	128	128	8	4	–	$(1024, 2048)$	800
Qwen3-4B	RI-TBRM	1×10^{-6}	32	32	16	4	4	$(1024, 20480)$	400
Qwen3-4B	ReVal	1×10^{-6}	32	32	16	4	–	$(1024, 20480)$	400
Qwen3-4B	TBRM	1×10^{-6}	32	32	16	4	–	$(1024, 20480)$	400
Qwen3-4B	GRPO	1×10^{-6}	32	32	16	4	–	$(1024, 20480)$	400
Qwen3-8B	RI-TBRM	2×10^{-6}	32	32	16	1	2	$(1024, 20480)$	400
Qwen3-8B	ReVal	2×10^{-6}	32	32	16	1	–	$(1024, 20480)$	400
Qwen3-8B	TBRM	2×10^{-6}	32	32	16	1	–	$(1024, 20480)$	400

the empirical comparisons, the backbone, training set, binary verifier, response-length limits, and evaluation protocol are aligned.

Offline reference-return annotation. RI-TBRM builds a static annotation cache from the frozen reference policy. For each training prompt, K independent reference responses define $\widehat{r}_{\text{ref}}(x)$ in Eq. (14), with $K = 8$, 4, and 2 for Qwen2.5-Math-7B, Qwen3-4B, and Qwen3-8B. The prompts and samples can be generated independently across inference workers. The cache stores the raw mean reward and reference log-partition, remains fixed throughout optimization, and is reusable by runs that share the reference checkpoint, dataset formatting, reward function, and sampling convention. Policy optimization then consumes these cached scalars together with fresh RL trajectories.

Reference-sample-size comparison. The main-table configuration uses $K = 8$ for Qwen2.5-Math-7B. Section 7.4 additionally reports $K = 16$ on AMC23, MATH500, and AIME25, together with the reported offline pre-sampling times of 27 minutes for $K = 8$ and 54 minutes for $K = 16$. These timings refer to the one-time pre-sampling stage, not the RL training loop. The additional $K = 16$ results cover only these three benchmarks; they are not used to replace the six-benchmark averages in Table 2. No repeated-run uncertainty estimates are available for this comparison.

Evaluation protocol. Following TBRM (Yuan et al., 2025b), we sample 32 responses per problem at temperature 1.0 on AIME24, AIME25, and AMC23. Avg@32 is the mean correctness over these responses, averaged across problems, rather than the fraction of problems solved by at least one response (pass@32). For MATH500 (Hendrycks et al., 2021), Minerva Math (Lewkowycz et al., 2022), and OlympiadBench (He et al., 2024), we use greedy decoding and report Avg@1. All methods within each backbone use the same prompt format, decoding configuration, stopping criteria, and answer verifier.

B Initialization Analysis and Proofs

B.1 Assumptions and notation

Unless stated otherwise, we condition on a fixed training prompt x and take response expectations under $\pi_{\text{ref}}(\cdot \mid x)$. Responses have a finite maximum horizon, prefix transitions are deterministic, and the outcome reward is a fixed function of the completed response. Terminal values are zero. Exact initialization means $\theta_0 = \theta_{\text{ref}}$ with the same forward-pass and probability conventions, so both logits and response probabilities initially coincide. Weight equality establishes the equality of the initial logits and their log-partitions.

The reference model and cached calibration statistics are fixed during differentiation. We differentiate the full residual while holding the sampled responses fixed. Gradient statements assume a common, parameter-independent response support and sufficient integrability to exchange differentiation and expectation. Finite vocabulary and horizon, positive softmax probabilities, and bounded

rewards provide one setting satisfying these assumptions. We use

$$\begin{aligned} Y(x,y) &= R(x,y)/\beta, \qquad & b(x) &= \mathbb{E}_{\mathrm{ref}}[Y \mid x], \\ \sigma^2(x) &= \mathrm{Var}_{\mathrm{ref}}(Y \mid x), & \widehat{b}_K(x) &= \widehat{r}_{\mathrm{ref}}(x)/\beta. \end{aligned} \tag{19}$$

The generic residual $\Delta_{\theta,v}$ and fixed-reference population loss $\mathcal{L}_v^{\mathrm{ref}}$ are defined in Eqs. (8) and (11). RI-TBRM sets $v = \widehat{b}_K$; $v = U_{\mathrm{ref}}$ and $v = 0$ recover the TBRM and ReVal trajectory residuals, respectively.

B.2 Complete-trajectory cancellation

For completeness, consider the general anchored values in Eq. (5). Their common translation commutes with log-sum-exp, so $Q_{\theta,v} - V_{\theta,v} = \log \pi_\theta$ at every nonterminal prefix. Setting the terminal value separately to zero gives

$$\begin{aligned} &\sum_{h=1}^{H_y} \left[Q_{\theta,v}(s_h,a_h) - \frac{r(s_h,a_h)}{\beta} - \log \pi_{\mathrm{ref}}(a_h \mid s_h) - V_{\theta,v}(s_{h+1}) \right] \\ &\quad = \sum_{h=1}^{H_y} \left[\log \frac{\pi_\theta(a_h \mid s_h)}{\pi_{\mathrm{ref}}(a_h \mid s_h)} - \frac{r(s_h,a_h)}{\beta} + V_{\theta,v}(s_h) - V_{\theta,v}(s_{h+1}) \right] \\ &\quad = V_{\theta,v}(x) + D_\theta(x,y) - Y(x,y) \\ &\quad = U_\theta(x) - U_{\mathrm{ref}}(x) + v(x) + D_\theta(x,y) - Y(x,y). \end{aligned} \tag{20}$$

Intermediate values telescope; token log-probabilities sum to the complete-response log-probability, and token rewards sum to R. The fixed translation therefore contributes only once through the starting value, even though it appears at every nonterminal prefix. The *trajectory* functional squares the sum of residuals. The shared offset anchors the initial prompt value and is applied consistently to all nonterminal prefixes.

B.3 Optimal scalar initialization of the residual

Proof of Proposition 1. Since $U_{\theta_0} = U_{\mathrm{ref}}$ and $D_{\theta_0} = 0$, Eq. (8) reduces to $\Delta_{\theta_0,v} = v - Y$. Its conditional mean is $v - b$. Expanding around b and using $\mathbb{E}[Y - b \mid x] = 0$ gives

$$\mathbb{E}[(v-Y)^2 \mid x] = (v-b)^2 - 2(v-b)\mathbb{E}[Y-b \mid x] + \mathbb{E}[(Y-b)^2 \mid x] = (v-b)^2 + \sigma^2. \tag{21}$$

The nonnegative squared-bias term vanishes only at $v = b$. □

For unshifted TBRM and zero anchoring, the initial risks are $\sigma^2 + (U_{\mathrm{ref}} - b)^2$ and $\sigma^2 + b^2$, respectively. Exact calibration removes the squared conditional bias and retains the return variance: $\mathrm{Var}_{\mathrm{ref}}(\Delta_{\theta_0,v} \mid x) = \sigma^2$ for every fixed v. Equation (10) identifies b with the reference policy's scaled value, rather than the value of arbitrary logits.

Why calibration is prompt-dependent. For a single global scalar v, averaging the risk over prompts gives

$$\mathbb{E}_x \sigma^2(x) + \mathrm{Var}_x\, b(x) + (v - \mathbb{E}_x b(x))^2. \tag{22}$$

Even the optimal global choice retains the between-prompt term $\mathrm{Var}_x\, b(x)$. This compares exact global and prompt-wise anchors; a finite-sample prompt-wise estimate incurs the additional calibration error below.

B.4 Initial expected-gradient decomposition

For a fixed prompt, define

$$\begin{aligned} g(x) &= \nabla_\theta U_\theta(x)\big|_{\theta_0}, \\ S(x,y) &= \nabla_\theta \log \pi_\theta(y \mid x)\big|_{\theta_0}, \\ q(x) &= \nabla_\theta \mathbb{E}_{y \sim \pi_\theta(\cdot|x)}[Y(x,y)]\big|_{\theta_0}. \end{aligned} \tag{23}$$

The distribution in $\mathcal{L}_v^{\mathrm{ref}}$ is fixed, whereas the expectation defining q varies with the policy.

Proof of Proposition 2. Suppress x. Normalization and the score identity give

$$\mathbb{E}_{\mathrm{ref}} S = \nabla_\theta \sum_y \pi_\theta(y \mid x)\Big|_{\theta_0} = 0, \qquad q = \mathbb{E}_{\mathrm{ref}}[YS]. \tag{24}$$

Because the reference quantities and v are frozen, $\nabla_\theta \Delta_{\theta,v}|_{\theta_0} = g + S$. Full differentiation therefore gives

$$\begin{aligned} \nabla_\theta \mathcal{L}_v^{\mathrm{ref}}(\theta_0 \mid x) &= \mathbb{E}_{\mathrm{ref}}[(v - Y)(g + S)] \\ &= (v - b)g + v\mathbb{E}_{\mathrm{ref}} S - \mathbb{E}_{\mathrm{ref}}[YS] = (v - b)g - q. \end{aligned} \tag{25}$$

Finally, $J_\beta(\pi_\theta \mid x)/\beta = \mathbb{E}_{\pi_\theta} Y - D_{\mathrm{KL}}(\pi_\theta \| \pi_{\mathrm{ref}})$. At θ_0, the KL derivative is $\mathbb{E}_{\mathrm{ref}}[(1 + D_{\theta_0})S] = \mathbb{E}_{\mathrm{ref}} S = 0$, proving $q = \beta^{-1} \nabla_\theta J_\beta(\pi_\theta \mid x)|_{\theta_0}$. □

Expected-gradient quantity. Proposition 2 characterizes the expected Euclidean loss gradient at reference-weight initialization under fixed reference-policy sampling. Its sample-level counterpart is $(v - Y)(g + S)$, and the optimizer then applies the clipping, adaptive preconditioning, and weight decay specified in Appendix A.

B.5 Finite-rollout calibration and sampling independence

Proof of Corollary 3. For iid reference calibration returns $Y_1, \ldots, Y_K$, $\widehat{b}_K = K^{-1} \sum_k Y_k$ satisfies $\mathbb{E}_{\mathrm{cal}} \widehat{b}_K = b$ and $\mathrm{Var}_{\mathrm{cal}}(\widehat{b}_K) = \sigma^2/K$. Conditional on a realized calibration set, an independent response has initial risk $\sigma^2 + (\widehat{b}_K - b)^2$ by Proposition 1. Averaging over calibration proves Eq. (17). Applying Proposition 2 to the fixed realized anchor gives

$$\begin{aligned} G_{\widehat{b}_K} &= (\widehat{b}_K - b)g - q, \quad \mathbb{E}_{\mathrm{cal}} G_{\widehat{b}_K} = -q, \\ \mathbb{E}_{\mathrm{cal}} \| G_{\widehat{b}_K} + q \|^2 &= \frac{\sigma^2}{K} \|g\|^2. \end{aligned} \tag{26}$$

The mean and squared deviation follow from the sample-mean moments. □

Across independent annotation sets, the conditional expected-gradient deviation has mean square $(\sigma^2/K)\|g\|^2$. For a realized cache, the initial conditional residual mean is $\widehat{b}_K - b$, and the population anchor gives exact conditional centering.

When finite calibration improves initial risk. Compared with any deterministic anchor $v(x)$, the expected initial risk is smaller exactly when

$$\frac{\sigma^2(x)}{K} < (v(x) - b(x))^2. \tag{27}$$

For binary rewards with reference success probability $p(x) > 0$, $b = p/\beta$ and $\sigma^2 = p(1 - p)/\beta^2$. Against zero anchoring, the condition is $K > (1 - p)/p$, equivalently $p > 1/(K + 1)$. At $p = 0$, both starts have zero initial residual. The condition maps reference success probability and reference sample size directly to expected initial residual risk.

Why calibration responses are kept separate. If the same K responses define $\bar{Y} = K^{-1} \sum_i Y_i$ and form the first update batch, the initial empirical gradient is $\widehat{G}_K = K^{-1} \sum_i (\bar{Y} - Y_i)(g + S_i)$. The g terms cancel, and independence across indices gives $\mathbb{E}[\bar{Y} S_i] = K^{-1} \mathbb{E}[Y_i S_i] = q/K$. Consequently,

$$\mathbb{E}\widehat{G}_K = -\left(1 - \frac{1}{K}\right) q, \qquad \mathbb{E}\left[\frac{1}{K} \sum_i (\bar{Y} - Y_i)^2\right] = \left(1 - \frac{1}{K}\right) \sigma^2. \tag{28}$$

The second equality follows by expanding $K^{-1} \sum_i Y_i^2 - \bar{Y}^2$ and using the sample-mean variance. At $K = 1$, both quantities vanish on the reused response. These self-inclusion factors arise from

sharing the sample in the mean and the update. Algorithm 1 uses independent annotation and update responses, giving the moments in Eq. (17).

B.6 Objective identities and target consistency

Connection to reference subtraction and reward centering. The zero-anchored residual is

$$\Delta_{\theta,0}(x,y) = U_\theta(x) - U_{\mathrm{ref}}(x) + D_\theta(x,y) - R(x,y)/\beta, \tag{29}$$

matching ReVal's reference-subtracted residual under the same reward and probability conventions (Wang et al., 2026). RI-TBRM can equivalently be written as

$$\Delta_\theta^{\mathrm{RI}}(x,y) = U_\theta(x) - U_{\mathrm{ref}}(x) + D_\theta(x,y) - \frac{R(x,y) - \widehat{r}_{\mathrm{ref}}(x)}{\beta}. \tag{30}$$

This identity expresses the reference-return anchor as a fixed, prompt-dependent reward translation of the reference-subtracted residual. The offset preserves the induced policy at fixed θ while changing the residual and its gradient.

Invariance to prompt-wise reward translations. Let $R'(x,y) = R(x,y) + k(x)$ for all responses to a prompt. Using the same calibration responses gives $\widehat{r}'_{\mathrm{ref}} = \widehat{r}_{\mathrm{ref}} + k(x)$, so $R' - \widehat{r}'_{\mathrm{ref}} = R - \widehat{r}_{\mathrm{ref}}$. Equation (30) is unchanged for every θ, even at finite K. If all responses to a prompt have the same reward, calibration is exact and the initial residual vanishes response by response. Its squared-loss gradient is then zero. Any decoupled weight-decay term is applied separately by the optimizer.

Proposition 4 (Full-support zero-residual consistency)**.** *Fix a prompt and a scalar anchor $v(x)$. Suppose the response distributions have common support with positive reference probability and $Z(x) := \mathbb{E}_{\mathrm{ref}}[e^{R(x,y)/\beta} \mid x] < \infty$. If $\Delta_{\theta,v}(x,y) = 0$ for every response in that support, then*

$$\pi_\theta(y \mid x) = \pi^*(y \mid x) := \frac{\pi_{\mathrm{ref}}(y \mid x) e^{R(x,y)/\beta}}{Z(x)}, \qquad V_{\theta,v}(x) = \log Z(x). \tag{31}$$

The policy π^* maximizes the original KL-regularized objective over response distributions on that support, independently of the numerical anchor.

Proof. Zero residual implies $\log[\pi_\theta(y \mid x)/\pi_{\mathrm{ref}}(y \mid x)] = R(x,y)/\beta - V_{\theta,v}(x)$. Exponentiating and normalizing over responses gives $e^{V_{\theta,v}(x)} = Z(x)$ and Eq. (31). For any supported response distribution π,

$$J_\beta(\pi \mid x)/\beta = \log Z(x) - D_{\mathrm{KL}}(\pi(\cdot \mid x) \| \pi^*(\cdot \mid x)). \tag{32}$$

Nonnegativity of KL divergence proves optimality. In the deterministic prefix process, a supported response distribution factors into its conditional token probabilities. □

The proposition identifies the common policy target of full-support zero-residual solutions across the fixed-anchor family.

C ROLLOUT REUSE WITH A REFRESHED REFERENCE

Reference-return anchoring specifies how a value baseline is tied to a reference policy's expected return; it does not require the same reference policy to remain fixed throughout training. Our default variant freezes the reference policy and estimates its returns using an independent calibration set. In this section, we consider an alternative implementation that refreshes the reference statistics once per outer training iteration and reuses the corresponding rollout responses for both return estimation and residual optimization. This removes the separate calibration rollout by changing both the reference schedule and the relation between calibration and training samples.

C.1 ALGORITHMIC FORMULATION

Reference selection and return estimation. Let θ_t denote the actor parameters before the optimizer updates of outer iteration t, and let $\pi_t = \pi_{\theta_t}$ denote the corresponding pre-update policy. We sample a prompt batch $\mathcal{B}_t$ and, for every $x \in \mathcal{B}_t$, generate $n > 1$ responses

$$y_{t,1}, \ldots, y_{t,n} \sim \pi_t(\cdot \mid x).$$

The resulting trajectory batch is

$$\mathcal{D}_t = \{(x, y_{t,i}) : x \in \mathcal{B}_t,\ i \in \{1, \ldots, n\}\}.$$

After evaluating $R_{t,i} = R(x, y_{t,i})$, we compute the prompt-wise empirical return

$$\bar{r}_t(x) = \frac{1}{n} \sum_{i=1}^{n} R_{t,i}. \tag{33}$$

Thus, the same responses used for policy optimization also estimate the return anchor. No additional reference responses or reward evaluations are required.

Before applying any optimizer update to $\mathcal{D}_t$, we perform one teacher-forced actor forward pass and cache, for every realized trajectory, its pre-update sequence log-probability $\log \pi_t(y_{t,i} \mid x)$ and its prompt log-partition $U_t(x)$. The sequence-level log-probability ratio used by the implementation is

$$\begin{aligned} D_{\theta,t}(x, y) &= \log \frac{\pi_\theta(y \mid x)}{\pi_t(y \mid x)} \\ &= \sum_{j=1}^{|y|} \left[\log \pi_\theta(y_j \mid x, y_{<j}) - \log \pi_t(y_j \mid x, y_{<j})\right]. \end{aligned} \tag{34}$$

Here,

$$U_\theta(x) = \log \sum_a \exp(z_\theta(a \mid x)) \tag{35}$$

is the log-partition of the logits predicting the first response token. Although the implementation stores this quantity for every trajectory, it depends on the prompt and pre-response context rather than on the sampled continuation.

Residual optimization with cached reference statistics. The implementation-level residual is

$$\widetilde{\Delta}_{\theta,t}(x, y) = \beta\left[U_\theta(x) - U_t(x) + D_{\theta,t}(x, y)\right] + \bar{r}_t(x) - R(x, y). \tag{36}$$

The corresponding objective is

$$\widetilde{\mathcal{L}}_t(\theta) = \frac{1}{|\mathcal{D}_t|} \sum_{(x,y) \in \mathcal{D}_t} \widetilde{\Delta}_{\theta,t}(x, y)^2. \tag{37}$$

Equivalently, defining the normalized residual

$$\Delta_{\theta,t}(x, y) = U_\theta(x) - U_t(x) + D_{\theta,t}(x, y) + \frac{\bar{r}_t(x) - R(x, y)}{\beta}, \tag{38}$$

gives $\widetilde{\Delta}_{\theta,t} = \beta\Delta_{\theta,t}$. The normalized and implementation-level objectives therefore differ only by a positive constant factor. We state Eq. (37) in its implementation-level form to make the connection to the training code explicit.

Gradients pass through both the current log-partition $U_\theta(x)$ and the current response log-probability $\log \pi_\theta(y \mid x)$. The sampled responses, rewards, and all cached pre-update quantities are treated as constants. At the refresh point $\theta = \theta_t$, both $U_\theta - U_t$ and $D_{\theta,t}$ vanish, and hence

$$\widetilde{\Delta}_{\theta_t,t}(x, y_{t,i}) = \bar{r}_t(x) - R_{t,i}. \tag{39}$$

Each response is therefore initially compared with the empirical mean reward of its own prompt group.

Update and reference-refresh schedule. The cached quantities are fixed for the entire optimization of $\mathcal{D}_t$, rather than refreshed after every optimizer mini-batch. In the reported Qwen2.5-Math-7B experiment, each outer iteration contains $|\mathcal{B}_t| = 64$ prompts and $n = 8$ responses per prompt, giving 512 trajectories. We perform one shuffled optimization epoch using mini-batches of eight prompts, or 64 trajectories, resulting in eight actor updates per outer iteration. The values $\bar{r}_t(x)$, $U_t(x)$, and $\log \pi_t(y_{t,i} \mid x)$ remain fixed across these updates.

Let θ_{t+1} denote the parameters after completing these mini-batch updates. The updated weights are then synchronized to the rollout workers, and the next outer iteration generates a new batch and caches a new reference snapshot. Thus, the reference statistics are fixed *within* an outer iteration and refreshed *between* outer iterations.

After the first mini-batch update, the remaining responses in $\mathcal{D}_t$ are no longer on-policy with respect to the intermediate actor parameters. The term "rollout reuse" refers to this reuse of a fixed trajectory batch across the updates of one outer iteration. It does not refer to the independent calibration responses used by the default frozen-reference variant.

The implementation uses a transfer queue to manage generated trajectories. The formulation and analysis below concern the zero-staleness case in which the selected trajectories were generated by the same pre-update policy π_t used to cache their reference statistics. If trajectories generated by an older policy version are retained in an asynchronous replay buffer, their behavior policy need not equal π_t, and the shared-sample identity below does not apply without an additional off-policy correction.

C.2 Calibration with shared samples

The same response contributes both to $\bar{r}_t(x)$ and to the residual being optimized. Consequently, the independent-calibration moments in Corollary 3 do not apply unchanged. Instead, the shared-sample calculation in Appendix B.5 describes the local gradient at each refresh point.

Conditional on x and θ_t, assume that $y_{t,1}, \ldots, y_{t,n}$ are iid samples from π_t, that sampling and scoring distributions match, and that the regularity conditions in Appendix B.1 hold. Define

$$Y_i = \frac{R_{t,i}}{\beta}, \qquad \bar{Y} = \frac{1}{n}\sum_{i=1}^{n} Y_i, \tag{40}$$

and

$$g_t = \nabla_\theta U_\theta(x)|_{\theta_t}, \qquad S_i = \nabla_\theta \log \pi_\theta(y_{t,i} \mid x)|_{\theta_t}, \qquad q_t = \mathbb{E}_{\pi_t}[Y_i S_i \mid x]. \tag{41}$$

Here, q_t is the gradient of the expected scaled reward at θ_t.

At the refresh point,

$$\widetilde{\Delta}_{\theta_t,t} = \beta(\bar{Y} - Y_i), \qquad \nabla_\theta \widetilde{\Delta}_{\theta,t}\Big|_{\theta_t} = \beta(g_t + S_i).$$

For one complete prompt group, define the normalized initial loss gradient

$$\widehat{G}_t = \frac{1}{2\beta^2}\, \nabla_\theta \left[\frac{1}{n}\sum_{i=1}^{n} \widetilde{\Delta}_{\theta,t}(x, y_{t,i})^2\right]\Bigg|_{\theta_t}. \tag{42}$$

Substitution gives

$$\widehat{G}_t = \frac{1}{n}\sum_{i=1}^{n}(\bar{Y} - Y_i)(g_t + S_i), \qquad \mathbb{E}[\widehat{G}_t \mid x, \theta_t] = -\left(1 - \frac{1}{n}\right) q_t. \tag{43}$$

The value-path contribution cancels because

$$\sum_{i=1}^{n}(\bar{Y} - Y_i) = 0.$$

The remaining expectation follows from the score identity $\mathbb{E}[S_i] = 0$ and $\mathbb{E}[\bar{Y} S_i] = q_t/n$. Therefore, for $n > 1$, the negative expected initial residual-loss gradient is a positive multiple of the local expected-reward gradient. The factor $1 - 1/n$ is the self-inclusion effect caused by using the same samples in both the empirical return and the residual. It is not an independent-calibration guarantee.

For $n = 1$, the response equals its own empirical reward baseline and the initial residual-loss gradient vanishes. These statements concern the full prompt-group gradient evaluated at the exact refresh point. They do not imply that every shuffled mini-batch gradient has the same cancellation property, nor do they characterize the later updates after the actor has moved away from θ_t.

Prompt-wise reward-translation invariance remains exact. Adding an arbitrary prompt-dependent constant $k(x)$ to every reward adds the same quantity to $\bar{r}_t(x)$ and $R(x, y)$, leaving Eq. (36) unchanged. However, because the reference snapshot changes between outer iterations, the resulting sequence of local objectives is not the single fixed-reference objective analyzed in Appendix B.6. The refreshed-reference variant preserves the local calibration principle without asserting an identical global optimization target.

C.3 Efficiency and evaluation

The sampling distinction is direct. Independent offline calibration draws additional responses solely to estimate the frozen reference policy's prompt-wise return. With reference refresh and rollout reuse, the rewards already associated with the training trajectories provide the required empirical return estimate. Consequently, no additional responses or reward evaluations are required for calibration.

The method is not free of reference-computation cost. Before the actor updates, it performs one teacher-forced forward pass over the reused trajectories to cache their pre-update log-probabilities and first-token log-partitions. It also recomputes the current quantities during actor optimization. Eliminating independent calibration sampling alone therefore does not establish a reduction in end-to-end wall-clock time.

Table 5: Qwen2.5-Math-7B with two reference schedules. AIME24, AIME25, and AMC23 use Avg@32; the remaining benchmarks use greedy Avg@1. Both variants use the same training-aligned chat template, answer-extraction procedure, and scoring implementation. All values are percentages. The frozen-cache row is reproduced from Table 2; the refreshed-reference row is a supplemental result rather than a matched-wall-clock comparison.

Variant	AIME24	AIME25	AMC23	MATH500	Minerva	Olympiad	Avg
Frozen cache ($K = 8$)	20.83	7.92	53.52	59.60	18.38	28.59	31.47
Refreshed reference	19.27	8.54	54.77	60.60	18.38	28.89	31.74

Table 5 reports the Qwen2.5-Math-7B results. The refreshed-reference variant obtains 27.53 across the three Avg@32 benchmarks and 35.96 across the three Avg@1 benchmarks. Its six-benchmark average is 31.74, compared with 31.47 for the default frozen-$K = 8$ cache, corresponding to a numerical increase of 0.27 percentage points. AIME24 is lower, Minerva Math is unchanged, and the remaining four benchmark scores are higher.

These results show that reference-return anchoring can retain competitive aggregate performance when the anchor is estimated from reused training rollouts rather than from a separate calibration stage. They do not establish uniform improvement, statistical significance across random seeds, or superiority under a matched wall-clock or generated-token budget. Independent pre-sampling is

therefore one way to construct a reference-return anchor, rather than a necessary sampling overhead of the anchoring principle.