EMO: A CLOSED-LOOP TEST-TIME COMPUTATION FRAMEWORK VIA ALTERNATING CONTEXT OPTIMIZATION AND LOGITS STEERING

Anonymous authors
Paper under double-blind review

ABSTRACT

Test-time context optimization adapts frozen language models through textual feedback, but ordinary conditional generation provides no separate control over how strongly to reinforce the distributional changes induced by a context update. We introduce **EMO**, a training-free framework that couples question-specific context optimization with logits steering. The E-step revises a system context using textual gradients from an external judge without access to gold answers. The M-step contrasts next-token distributions under the revised and preceding contexts at a shared decoding prefix, transforms this contrast into a steering vector, and applies a confidence-constrained logit correction. Feedback on the steered response drives the next context update, coupling adaptation across rounds with intervention within each generation. A logit-space analysis connects the steering vector to the context-induced logit displacement and identifies sufficient conditions for one-round target-KL improvement. Under greedy decoding across eight models and six benchmarks, EMO improves the zero-shot baseline in 39 of 40 aggregated settings (47 of 48 before merging the two AIME editions), averaging $+9.10$ percentage points over the 40 model–task-group pairs. With the same judge, it adds $+1.58$ points on average over context-only optimization, winning in 25 settings and tying in four, with MMLU-Pro gains across all eight models. At matched target-model generation budgets, accuracy advantages over best-of-N are strongest on Llama, while oracle coverage is higher in four of six comparisons despite fewer candidates. The coverage–selection gap motivates allocating test-time computation jointly to candidate generation and verification.

1 INTRODUCTION

Parameter updates and context refinement provide complementary routes to adapting a large language model (LLM). Parameter optimization, including fine-tuning, reinforcement learning, and preference alignment, incorporates task data or feedback into model weights (Hu et al., 2021; Shao et al., 2024; Rafailov et al., 2023b). For an individual test question, context optimization adapts a frozen model by using feedback to revise its textual instructions before generating a new response (Yuksekgonul et al., 2025; Zhang et al., 2025; Madaan et al., 2023). We study how to translate such question-specific feedback into explicit control of the decoding process.

A revised context already changes the next-token distribution at every prefix. Ordinary conditional generation, however, provides no separate control over the strength or token support of those changes. Comparing the distributions before and after a context update reveals how the revision changes token preferences at the current prefix. This motivates our central question: *how can feedback-driven context updates supply a controllable intervention on the generation they guide?*

We propose **EMO**, which alternates context optimization and distribution refinement for each question. The *E-step* revises a system context using textual gradients (Yuksekgonul et al., 2025) from an external judge without access to gold answers. The *M-step* compares the revised and preceding contexts at a shared generated prefix, constructs a steering vector from their distribution contrast, and applies a confidence-constrained correction to the revised context's logits. The resulting response is evaluated to provide feedback for the next E-step. Each context update thus determines a decoding

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.

intervention whose output informs the next update, closing the loop while model weights remain fixed.

A logit-space view explains the relationship between the two steps. Parameter and context updates both admit additive logit representations, while EMO uses the context-induced displacement to construct its additional steering correction. Context edits induce finite logit changes through a frozen model, whereas parameter updates also alter the model's response to subsequent contexts and prefixes. This representation links adaptation to decoding, while our one-round analysis identifies sufficient conditions for improvement toward a target distribution.

Across eight models and six benchmarks, EMO improves over Base in 39 of 40 aggregated settings, averaging $+9.10$ percentage points, with benefits spanning model scale, generation, family, and thinking mode. With the judge and candidate count held fixed, adding the M-step yields $+1.58$ points on average over context-only optimization and improves MMLU-Pro for all eight models. At matched target-model generation budgets, EMO also achieves higher oracle coverage than independent sampling in four of six settings despite producing fewer candidates, indicating benefits in candidate generation alongside improvements in selected accuracy.

Our contributions are summarized as follows:

- **EMO: coupled context optimization and steering.** A training-free framework that turns successive context updates into prefix-dependent decoding interventions and uses the steered responses to drive further context adaptation.
- **Logit-space interpretation and conditional analysis.** We relate the context-induced logit displacement to the implemented steering vector (Eq. 5) and establish one-round target-KL improvement under context-improvement and faithful-steering assumptions (Theorem 1).
- **Empirical evidence across four model axes.** Broad improvements over Base and additional gains over context-only optimization, supported by component ablations, decoding comparisons, and generation-budget analyses that distinguish candidate coverage from answer selection.

2 Related Work

Parameter optimization. Parameter adaptation includes supervised fine-tuning and its parameter-efficient variants (Hu et al., 2021), reinforcement learning (Schulman et al., 2017; Shao et al., 2024; Guo et al., 2025; Yu et al., 2026), preference optimization (Rafailov et al., 2023a), and distillation (Snell et al., 2022; Agarwal et al., 2024). These approaches incorporate training signals into model weights. EMO operates on a frozen checkpoint and uses feedback on the current question to adapt its context and decoding, making it complementary to parameter adaptation.

Context optimization. A second line adapts textual inputs while keeping model weights frozen, including textual gradients and prompt optimization (Yuksekgonul et al., 2025; Yang et al., 2024), self-correction (Madaan et al., 2023; Shinn et al., 2023; Gou et al., 2024), and agentic context engineering that evolves skills and memories (Zhang et al., 2026; Ye et al., 2026). These approaches use textual revisions to influence subsequent generation. EMO's E-step builds on this idea; its M-step additionally uses the distributional change between successive contexts to control decoding at each prefix. The steered response then becomes the basis for further textual feedback.

Decoding-time distribution intervention. A third family intervenes on the output distribution at decoding time (Zhang et al., 2024; Li et al., 2023; Turner et al., 2024). Closely related task-aware approaches include CoS (He et al., 2025), SDA (Xia & Deng, 2026), SVDecode (Hu et al., 2026), and JiTRL (Li et al., 2026). EMO connects decoding-time intervention to per-question context optimization: each response informs the next context update, whose distribution contrast supplies the steering signal at every subsequent prefix. The logit-space view in Section 3.1 formalizes how these context updates determine prefix-dependent decoding interventions. Test-time compute can also be spent on parallel sampling, including self-consistency (Wang et al., 2022) and best-of-N with a verifier or judge (Cobbe et al., 2021; Brown et al., 2024), or sequential refinement (Snell et al., 2024). Our experiments compare EMO with context-only refinement and, separately, with SC/BoN at matched target-model generation budgets (Section 4.4).

3 METHOD

EMO couples context refinement and decoding interventions through feedback on generated responses. Context updates induce question-specific changes in token preferences; steering converts these changes into controlled logit corrections at each decoding prefix. We first relate the two updates in logit space, then describe the algorithm and the conditions for target alignment.

3.1 A UNIFIED VIEW: OPTIMIZATION AS LOGITS STEERING

Let $\xi = (\boldsymbol{\theta}, C)$ denote model parameters and system context jointly. The response distribution is $P_\xi(y \mid q) = \prod_t P_\xi(y_t \mid q, y_{<t})$, where q is a question and $y_t, y_{<t}$ denote token t and its prefix in response y. Numerical vectors use bold lowercase and matrices bold uppercase; scalars are italic. Distributions retain uppercase symbols, interpreted as probability columns in matrix operations.

A shared alignment objective. SFT minimizes reference-response negative log-likelihood (NLL) over $\boldsymbol{\theta}$ with C fixed. TextGrad (Yuksekgonul et al., 2025) instead refines text from loss feedback with $\boldsymbol{\theta}$ frozen; when instantiated with reference NLL, it targets the same objective through C.

Lemma 1 (Unified Objective: NLL–KL Equivalence)**.** *With one reference response per question,*

$$\mathcal{L}(\xi) = \mathbb{E}_{(q,y^\star)\sim\mathcal{D}}[-\log P_\xi(y^\star \mid q)] = \mathbb{E}_{q\sim\mathcal{D}}\mathrm{KL}\Big(\hat{P}_{\mathrm{target}}(\cdot \mid q) \,\|\, P_\xi(\cdot \mid q)\Big), \tag{1}$$

where $\mathcal{D}$ is the empirical distribution of question–reference pairs, $y^\star = y^\star(q)$ is the reference, $\mathcal{L}$ is the mean NLL, and $\hat{P}_{\mathrm{target}}(\cdot \mid q) = \delta_{y^\star(q)}$ is a point mass at that reference.

Only the reference contributes to the point-mass KL (Appendix A.1). At test time, judge feedback provides a surrogate signal for this unavailable reference-based objective; textual gradients encode natural-language criticism rather than derivatives of strings.

A shared execution form. Consider a shared history $h = (q, y_{<t})$, with logits $\mathbf{z}$, model conditional $P = \mathrm{Softmax}(\mathbf{z})$, and target conditional $Q = \hat{P}^t_{\mathrm{target}}(\cdot \mid h)$.

Lemma 2 (Unified Steering Form)**.** *Any parameter or context update $\xi \to \xi^+$ induces an exact additive correction $\Delta\mathbf{z} = \mathbf{z}_{\xi^+} - \mathbf{z}_\xi$, yielding $\mathrm{Softmax}(\mathbf{z}_\xi + \Delta\mathbf{z})$ at each history. In unrestricted logit space, gradient descent on the conditional target KL gives*

$$\hat{\mathbf{z}} = \mathbf{z} + \eta\boldsymbol{\delta}^*, \qquad \boldsymbol{\delta}^* = \mathbf{J}_P \frac{Q}{P} = Q - P, \qquad \mathbf{J}_P = \mathrm{diag}(P) - PP^\top, \tag{2}$$

where $\hat{\mathbf{z}}$ contains updated logits, $\eta > 0$ is an analysis step size, $\boldsymbol{\delta}^*$ is the ideal descent direction, and $\mathbf{J}_P$ is the softmax Jacobian. Division is componentwise; $\mathrm{diag}(P)$ is the diagonal probability matrix. If $Q \neq P$, every sufficiently small $\eta > 0$ decreases the conditional KL.

Proof sketch. The softmax chain rule gives $\nabla_{\mathbf{z}}\mathrm{KL}(Q\|P) = P - Q$, so the derivative along $\boldsymbol{\delta}^*$ is $-\|Q - P\|_2^2 < 0$. The additive realization follows from the logit difference (Appendix A.2).

Parameter gradients propagate through the model Jacobian, whereas textual rewrites induce finite logit differences; the resulting update directions need not coincide. Since Q is unavailable, EMO constructs a computable direction from the context-induced distribution contrast.

3.2 EMO: ALTERNATING CONTEXT OPTIMIZATION AND LOGITS STEERING

EMO alternates **context estimation (E-step)** and **distribution refinement (M-step)** per question. The EM analogy is structural: textual feedback is not a posterior expectation, and steering does not maximize a specified common likelihood.

3.2.1 E-STEP: CONTEXT OPTIMIZATION VIA TEXTUAL GRADIENTS

Initialize $R_0 = \mathrm{Generate}(\boldsymbol{\theta}, C_0, q)$, where C_0 is reset for each question and $\mathrm{Generate}$ uses ordinary target-model decoding. Judge feedback guides the context update:

$$C_{k+1} = \mathrm{Opt}\Big(C_k, g_C^{(k)}\Big), \qquad g_C^{(k)} = \mathrm{Backward}(C_k, q, R_k, F_k), \tag{3}$$

where k indexes rounds, R_k is the current response, and F_k is feedback from a judge without gold-answer access. $\mathrm{Backward}$ attributes response errors to C_k, producing the textual gradient $g_C^{(k)}$; Opt uses this natural-language guidance to construct C_{k+1}. Subsequent rounds use the preceding M-step's steered response as R_k. Rewrites update the domain role, reasoning strategies, pitfalls, and output requirements while keeping the question fixed. Feedback and rewrite prompts forbid final-answer disclosure; a pattern filter rejects suspected answer-bearing rewrites and retains C_k. Target-KL improvement is an analysis assumption (Section 3.3), not guaranteed by the textual update.

3.2.2 M-STEP: DISTRIBUTION CONTRAST AS LOGITS STEERING

At each decoding step, evaluate $P_{\mathrm{opt}} = P_{\boldsymbol{\theta}}(\cdot \mid C_{k+1}, q, y_{<t})$ and $P_{\mathrm{base}} = P_{\boldsymbol{\theta}}(\cdot \mid C_k, q, y_{<t})$ on the same generated prefix. Here "base" denotes C_k; round, question, and prefix indices are suppressed. Following SVDecode (Hu et al., 2026), the contrast yields:

$$\boldsymbol{\delta}_t = \mathbf{J}_{P_{\mathrm{opt}}} \log \frac{P_{\mathrm{opt}}}{P_{\mathrm{base}}} = \nabla_{\mathbf{z}_t^{\mathrm{opt}}} \mathrm{KL}(P_{\mathrm{opt}} \| P_{\mathrm{base}}), \tag{4}$$

where $\boldsymbol{\delta}_t$ is the raw steering vector, $\mathbf{z}_t^{\mathrm{opt}}$ contains logits under C_{k+1}, and $\mathbf{J}_{P_{\mathrm{opt}}}$ is the softmax Jacobian; differentiation holds P_{base} fixed. Logarithms and ratios act componentwise. For nonzero $\boldsymbol{\delta}_t$, an unmasked positive step locally increases divergence from the preceding-context distribution; this alone does not ensure target alignment.

From context updates to steering. At a shared prefix, context updates give:

$$\Delta \mathbf{z}_t = \mathbf{z}_t^{\mathrm{opt}} - \mathbf{z}_t^{\mathrm{base}}, \qquad \boldsymbol{\delta}_t = \mathbf{J}_{P_{\mathrm{opt}}} \Delta \mathbf{z}_t, \tag{5}$$

where $\mathbf{z}_t^{\mathrm{base}}$ contains logits under C_k and $\Delta \mathbf{z}_t$ is the context-induced displacement. This identity follows because $\log(P_{\mathrm{opt}}/P_{\mathrm{base}})$ differs from $\Delta \mathbf{z}_t$ only by a constant across tokens, which the softmax Jacobian annihilates. The Jacobian centers and probability-weights the displacement from Lemma 2; sharing the prefix isolates the context change. A token receives a positive raw correction only when its log-ratio exceeds the P_{opt}-weighted average. This centered expression can be computed in time linear in vocabulary size, without forming a dense Jacobian or backpropagating through model parameters (Appendix A.3).

Confidence-constrained decoding. To restrict the contrast signal to tokens supported by the optimized context, set $\mathbb{I}_t(v) = \mathbf{1}\{P_{\mathrm{opt}}(v) \geq \alpha \max_{v'} P_{\mathrm{opt}}(v')\}$ and apply

$$\hat{\delta}_t(v) = \mathbb{I}_t(v)\delta_t(v) + (1 - \mathbb{I}_t(v))\lambda, \qquad \alpha \in (0, 1], \quad \lambda = -100, \tag{6}$$

where v, v' are tokens, $\mathbb{I}_t$ indicates whether a token passes the relative threshold α, λ penalizes exclusion, and $\delta_t(v), \hat{\delta}_t(v)$ are entries of $\boldsymbol{\delta}_t, \hat{\boldsymbol{\delta}}_t$. Increasing α can only shrink the retained set, which always includes a maximizer; excluded tokens receive finite penalties. The constrained vector then corrects the logits used for greedy decoding or sampling:

$$\tilde{\mathbf{z}}_t = \mathbf{z}_t^{\mathrm{opt}} + \mu \hat{\boldsymbol{\delta}}_t, \qquad y_t = \mathrm{Decode}(\mathrm{Softmax}(\tilde{\mathbf{z}}_t)), \tag{7}$$

where $\tilde{\mathbf{z}}_t$ contains corrected logits, $\mu \geq 0$ scales the constrained vector, and Decode selects y_t greedily or by sampling. The defaults are $(\mu, \alpha) = (2.4, 0.1)$; excluded tokens receive the additive correction $\mu\lambda$. At $\mu = 0$, the method reduces to context-only optimization. Both contrast and masking are recomputed at each prefix, before sampling temperature or truncation is applied.

Closed-loop execution. Algorithm 1 performs K optimization rounds, generating at most T tokens per response and stopping at the end-of-sequence (EOS) token. It returns the earliest response maximizing the stored score $a_i = \mathcal{J}(q, R_i)$, where $\mathcal{J}$ is the judge and $i \in \{0, \ldots, K\}$ indexes candidates. Two context-conditioned streams maintain separate key–value caches while sharing the generated prefix. Thus $K = 4$ yields five candidates and nine target-model generation equivalents; this excludes judge/rewrite costs and response-length differences and does not imply equal latency.

3.3 THEORETICAL ANALYSIS

Under context-improvement and faithful-steering assumptions, we obtain a local guarantee of improvement in target sequence KL.

Assumption 1 (TextGrad Oracle)**.** *The updated context C_{k+1} has strictly smaller target sequence KL than C_k (Eq. (17), Appendix A.4).*

Assumption 2 (Faithful Constrained Steering)**.** *For fixed α, the masked direction $\hat{\boldsymbol{\delta}}_t$ has positive aggregate inner product with $Q_t - P_{\text{opt}}$, summed over positions and averaged over target prefixes, where Q_t is the target conditional (Eq. (18), Appendix A.4).*

Theorem 1 (Conditional One-Round Improvement)**.** *Fix a question, a finite vocabulary and horizon, finite logits, and α. Under Assumptions 1–2, every sufficiently small $\mu > 0$ yields a steered sequence distribution with smaller target KL than both $P_{\boldsymbol{\theta},C_{k+1}}$ and $P_{\boldsymbol{\theta},C_k}$.*

The autoregressive KL chain rule proves the result (Appendix A.4). It concerns full softmax distributions before temperature scaling or truncation, and compares against unsteered context distributions; it guarantees neither monotonic accuracy nor improvement between steered rounds. Judge selection preserves only the best stored judge score (Appendix A.5).

Algorithm 1 EMO

Require: Model $\boldsymbol{\theta}$, question q, context C_0, judge $\mathcal{J}$, rounds K, μ, α

$R_0 \leftarrow \text{Generate}(\boldsymbol{\theta}, C_0, q)$
$a_0 \leftarrow \mathcal{J}(q, R_0)$
for $k = 0, \ldots, K-1$ **do**
  $F_k \leftarrow \text{Feedback}_{\mathcal{J}}(q, R_k)$
  $g_C^{(k)} \leftarrow \text{Backward}(C_k, q, R_k, F_k)$
  $C_{k+1} \leftarrow \text{Opt}(C_k, g_C^{(k)})$
  Retain C_k if the rewrite is rejected
  Initialize empty prefix $y_{<1}$
  for $t = 1, \ldots, T$ **do**
    Evaluate $P_{\text{opt}}, P_{\text{base}}$ at $y_{<t}$
    Compute $\boldsymbol{\delta}_t$ by Eq. (4)
    Apply α-mask by Eq. (6)
    $\tilde{\mathbf{z}}_t \leftarrow \mathbf{z}_t^{\text{opt}} + \mu\hat{\boldsymbol{\delta}}_t$
    $y_t \leftarrow \text{Decode}(\text{Softmax}(\tilde{\mathbf{z}}_t))$
    Append y_t; stop on EOS
  end for
  $R_{k+1} \leftarrow$ generated response
  $a_{k+1} \leftarrow \mathcal{J}(q, R_{k+1})$
end for
return earliest highest-scoring R_i

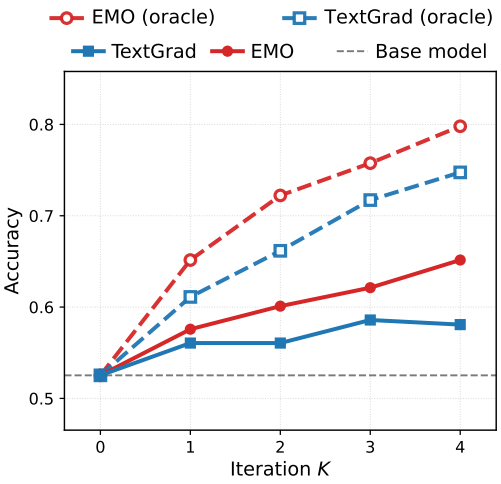


Figure 1: Accuracy and oracle coverage versus iteration on Qwen3-14B/GPQA-Diamond. Solid markers show judge-selected accuracy; hollow markers show oracle coverage over $K+1$ candidates, including the initial response. These trajectories are empirical.

4 Experiments

4.1 Experimental Setup

Models and benchmarks. We evaluate eight models spanning 3B–32B parameters and two families: Qwen3-8B/14B/32B, Qwen2.5-7B, Qwen3.5-9B, DeepSeek-R1-0528-Qwen3-8B, Llama-3.1-8B-Instruct, and Llama-3.2-3B-Instruct, covering both thinking and non-thinking models (Appendix B.2). We use six benchmarks covering multiple-choice and free-form formats: AIME-2025, AIME-2026, GPQA-Diamond, GSM8K, MATH-500, and MMLU-Pro (560 questions: 40 per category, sampling seed 42). The main table averages the two AIME editions into one of five equally weighted task groups (Appendix B.1).

Judge and scoring. The external judge, Qwen3.5-122B-A10B (FP8), assigns scores from 0 to 10 without access to gold answers; feedback prompts forbid final-answer disclosure. Judge scoring is deterministic, while feedback and rewriting use temperature 0.7, so optimization remains stochastic even with greedy target decoding (Appendix B.3). Reported accuracy uses task-specific answer extraction and gold-based correctness rules, independently of judge scores. Under *judge-select*, the highest-scoring response among the initial generation and K optimization rounds is selected, with ties resolved in favor of the earliest response. We separately report *oracle coverage*: the fraction of questions with at least one correct response in the candidate pool.

Table 1: **Main results.** Accuracy (%) across 8 models and 6 benchmarks (AIME-2025/2026 merged into one task group), greedy decoding. Each group reports the zero-shot baseline (Base), context-only optimization (TextGrad), and EMO; the best per group is bolded.

	AIME (%)			GPQA-D (%)			GSM8K (%)			MATH-500 (%)			MMLU-Pro (%)		
Model	Base	TG	EMO	Base	TG	EMO	Base	TG	EMO	Base	TG	EMO	Base	TG	EMO
Llama-3.2-3B	0.0	**5.0**	3.3	26.8	**40.4**	34.3	71.9	86.5	**87.1**	39.8	**61.0**	60.4	34.8	45.7	**53.0**
Qwen2.5-7B	5.0	11.7	**16.7**	32.8	42.9	**45.5**	87.8	90.7	**91.7**	70.0	78.0	**79.2**	54.8	61.6	**65.5**
Llama-3.1-8B	0.0	**3.3**	**3.3**	29.3	**43.9**	38.9	82.1	**89.6**	89.4	43.2	65.4	**65.6**	45.4	53.9	**63.9**
Qwen3-8B	20.0	43.3	**46.7**	49.0	54.5	**56.6**	91.4	**92.7**	92.3	75.0	82.0	**83.0**	63.6	64.8	**67.5**
DR1-Qwen3-8B	8.3	**18.3**	16.7	46.0	**48.0**	**48.0**	82.2	**88.1**	87.9	56.2	**66.6**	64.0	62.0	67.0	**68.2**
Qwen3.5-9B	25.0	26.7	**45.0**	39.4	41.9	**45.5**	89.0	**90.6**	**90.6**	48.2	50.0	**50.4**	62.9	63.6	**65.0**
Qwen3-14B	20.0	**33.3**	**33.3**	52.5	58.1	**65.2**	91.7	**92.0**	91.5	76.2	83.2	**83.8**	71.4	73.6	**75.2**
Qwen3-32B	18.3	43.3	**48.3**	59.6	**64.6**	62.1	91.4	91.9	**92.0**	78.6	83.8	**86.0**	71.4	72.3	**74.3**
Average	12.1	23.1	**26.7**	41.9	49.3	**49.5**	85.9	90.3	**90.3**	60.9	71.3	**71.6**	58.3	62.8	**66.6**

Baselines and protocol. We compare **Base** (zero-shot decoding) and **TextGrad** (our E-step without the M-step), adapting TextGrad from task-level to per-question optimization to match our protocol. TextGrad and EMO share the initial context, baseline response, judge, and $K = 4$ optimization rounds, with judge-select over five candidates. EMO uses fixed defaults $\mu = 2.4$ and $\alpha = 0.1$ (Appendix B.5). Main results use greedy target decoding; budget comparisons use temperature 0.6, top-p 0.95, and seeds $\{0, 1, 2\}$ against self-consistency (SC) and best-of-N (BoN) (Section 4.4). Full decoding settings and token limits appear in Appendix B.3.

4.2 Main Results

Broad gains across the model grid. Table 1 separates overall gains over Base from incremental gains over context-only optimization. **EMO improves over Base in 39 of 40 cells** (47 of 48 before merging the two AIME editions), averaging $+9.10$ percentage points across the 40 model–task-group pairs. All eight models improve on average; EMO also exceeds TextGrad in 25 cells and ties in four, with a mean increment of $+1.58$ points. Using Qwen3-8B as a common reference, we organize the analysis along four axes: model scale, generation, family, and thinking mode.

Scale: complementing model scaling. We first test whether adaptation benefits persist as target models scale. Qwen3-14B with EMO exceeds Qwen3-32B Base on all five task groups, averaging 69.80 versus 63.86. Likewise, optimized Llama-3.2-3B exceeds Llama-3.1-8B Base on every task group. Larger models also retain substantial gains: Qwen3-32B improves by 8.68 points on average and 30.0 on AIME. Accuracy headroom alone is insufficient to predict improvement: Qwen3 gains little on near-saturated GSM8K, yet Llama gains little on AIME despite near-zero baselines. Mean gains across Qwen3-8B/14B/32B are non-monotonic ($9.42/7.44/8.68$): model size alone does not determine adaptation benefit. Adaptation can close size-related gaps, but its benefit depends on whether feedback-guided generation can reach correct solutions.

Generation: sustained gains as models evolve. Across Qwen2.5-7B, Qwen3-8B, and Qwen3.5-9B, EMO yields mean gains of 9.64, 9.42, and 6.40 points over Base. The incremental gains over TextGrad are 2.74, 1.76, and 4.74 points, respectively: the newest generation has the smallest overall gain but the largest M-step increment. On Qwen3.5-9B, context-only optimization gains just 1.66 points on average, whereas the full method gains 6.40. AIME makes this contrast concrete: Base, TextGrad, and EMO score 25.0, 26.7, and 45.0. Thus, limited gains from context rewrites need not imply limited scope for adaptation: steering can further exploit the distributional changes induced by those rewrites.

Family: narrowing gaps at matched model size. Qwen3-8B and Llama-3.1-8B allow a cross-family comparison at matched parameter scale. On MMLU-Pro, their accuracy gap shrinks from 18.2 points at Base to 3.6 after applying EMO to both models. The corresponding gaps shrink from 31.8 to 17.4 on MATH-500 and from 9.3 to 2.9 on GSM8K. Steering contributes directly to the MMLU-Pro result: its increment over TextGrad is 10.0 points for Llama versus 2.7 for Qwen.

Table 2: **Component ablation** on Qwen3-8B (accuracy %, greedy). "w/o M-step" reduces to context-only optimization (TextGrad); "w/o Judge" replaces the external judge with self-judging; "w/o Both" combines these changes.

Dataset	Base	EMO	w/o M-Step	w/o Judge	w/o Both
AIME	20.0	**46.7**	43.3	31.7	25.0
GPQA-D	49.0	**56.6**	54.5	51.0	54.5
GSM8K	91.4	92.3	**92.7**	91.8	92.4
MATH-500	75.0	**83.0**	82.0	78.6	78.0
MMLU-Pro	63.6	**67.5**	64.8	65.2	63.7
$\overline{\Delta}$	–	**+9.4**	+7.7	+3.9	+2.9

Table 3: **Decoding-strategy ablation.** EMO gains over Base (percentage points; single-seed results, seed 0) under greedy, top-p (temperature 0.6, $p = 0.95$), and top-k ($k = 20$).

Dataset	Model	Greedy	top-p	top-k
GPQA-D	Qwen3-8B	+7.6	+4.5	+6.6
	Llama-3.1-8B	+9.6	+10.1	+12.1
MATH-500	Qwen3-8B	+8.0	+5.6	+4.4
	Llama-3.1-8B	+22.4	+20.6	+21.6
MMLU-Pro	Qwen3-8B	+3.9	+6.1	+5.7
	Llama-3.1-8B	+18.6	+21.8	+25.2

On Llama/MATH-500, however, TextGrad already reaches 65.4 versus EMO's 65.6. The narrowed family gaps therefore reflect task-dependent contributions from context refinement and steering.

Thinking mode: gains with and without explicit reasoning modes. The non-thinking Qwen2.5-7B, Llama-3.1-8B, and Llama-3.2-3B gain 9.64, 12.22, and 12.96 points on average. Qwen2.5-7B also exceeds TextGrad on all five task groups, showing that M-step gains do not require a native thinking mode. Models with thinking modes also improve: the Qwen3 series gains 7.44–9.42 points, and the reasoning-distilled DR1-Qwen3-8B gains 6.02. DR1 improves over Base on all five task groups, showing that reasoning distillation leaves scope for question-specific adaptation. Yet its M-step increment is -0.64 points on average, with an additional gain only on MMLU-Pro (68.2 vs. 67.0). DR1 and Qwen3.5-9B therefore have similar overall gains (6.02 vs. 6.40) but contrasting M-step contributions (-0.64 vs. $+4.74$). One possible explanation is that distillation reshapes context-conditioned distributions and their sensitivity to logit interventions, altering the benefit of a fixed steering rule. Establishing this mechanism requires matched pre-/post-distillation controls.

Where the M-step is most effective. Across these axes, **MMLU-Pro provides the most consistent steering gains**: all eight models exceed TextGrad, by 3.76 points on average. MMLU-Pro oracle coverage also improves over TextGrad for all eight models (mean $+9.6$), whereas mean oracle increments on MATH-500 and GSM8K are at most 1.4 points (Appendix B.4.2). At equal candidate counts, these results associate steering gains with broader coverage; mixed GPQA-D results show that multiple-choice format alone does not explain the effect. Figure 1 shows how the advantage develops across rounds on Qwen3-14B/GPQA-D. From a shared 52.5 baseline, EMO continues improving to 65.2, while TextGrad levels off near 56–59 and ends at 58.1. EMO's oracle curve also remains above TextGrad's after the first update, reaching 79.8 at $K = 4$. Because oracle coverage does not depend on final answer selection, the curves indicate that improved candidate generation contributes to the 7.1-point accuracy advantage. We next analyze components, generation budgets, and selection (Section 4.5).

4.3 Component Ablations and Robustness

External judging supplies the larger average contribution. Table 2 varies M-step inclusion and judge choice jointly to distinguish their contributions on Qwen3-8B. With the M-step retained, replacing self-judging with the external judge raises average accuracy by 5.56 points; without the M-step, the increase is 4.74 points. With steering, the external judge's largest advantages are on AIME (46.7 vs. 31.7) and GPQA-D (56.6 vs. 51.0). Since judge choice affects both feedback and response selection, these differences measure their combined effect.

Steering adds to context optimization under both judge choices. Holding judge choice fixed, adding the M-step improves average accuracy by 1.76 points with the external judge and 0.94 with self-judging. These contributions interact: on Qwen3-8B/GPQA-D, steering gains 2.1 points with the external judge but loses 3.5 with self-judging. On Qwen3-32B/GPQA-D, however, steering loses

Table 4: **Matched generation budgets** on MMLU-Pro (accuracy %, mean±std over 3 seeds, top-p sampling). iter/N: EMO rounds K / target-model generation equivalents $N = 2K + 1$. *Pass*: oracle coverage over $K + 1$ candidates for EMO and N for SC/BoN.

iter/N	SC	BoN	EMO	Pass (oracle)	
				SC/BoN	EMO
Qwen3-8B					
0/1	65.5±1.2	65.5±1.2	63.9±1.3	65.5±1.2	63.9±1.3
1/3	66.8±1.2	67.1±1.0	**67.9**±1.1	75.2±0.8	**75.8**±1.5
2/5	67.4±1.0	68.2±0.7	**69.4**±0.6	78.7±0.3	**80.3**±1.7
3/7	67.7±0.3	68.2±0.6	**70.0**±0.8	80.6±0.3	**83.0**±2.2
4/9	67.1±1.2	68.4±0.5	**70.9**±0.7	81.9±0.5	**84.6**±1.7
Llama-3.1-8B					
0/1	45.1±0.6	45.1±0.6	45.1±1.0	45.1±0.6	45.1±1.0
1/3	50.4±1.5	55.1±1.0	**59.4**±2.6	62.3±1.1	**66.9**±2.1
2/5	49.7±0.4	58.0±1.1	**63.7**±1.5	68.9±0.4	**76.2**±2.1
3/7	50.7±1.0	61.0±0.8	**65.7**±1.1	73.9±0.1	**80.2**±1.3
4/9	51.2±0.8	62.5±0.5	**68.2**±0.8	76.4±0.2	**83.2**±1.1

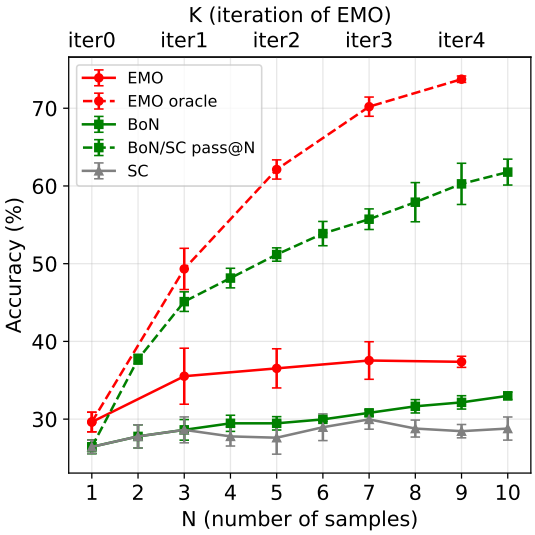


Figure 2: **Generation-budget scaling** on Llama-3.1-8B/GPQA-D (top-p; mean±std over three seeds). At $N = 2K + 1$, EMO has $K + 1$ candidates; SC/BoN have N. Dashed curves show oracle coverage.

2.5 under either judge (Appendix B.4.3). These results show that steering effectiveness depends on both feedback quality and the target model, rather than on judge choice alone.

Robustness across decoding settings. Table 3 tests whether gains depend on deterministic target decoding: EMO improves over Base in all 18 single-seed configurations spanning greedy, top-p, and top-k decoding. For Llama-3.1-8B on MMLU-Pro, the respective gains are $+18.6$, $+21.8$, and $+25.2$ points. The full method's gains persist under stochastic token selection; this comparison does not isolate the M-step under sampling.

Confidence-constrained execution. The hyperparameter sweep tests correction strength and retained token support separately. On Qwen3-8B/GPQA-D, accuracy stays at 54.5–56.6 for $\mu = 1.6$–3.2 at $\alpha = 0.1$. At $\mu = 2.4$, thresholds $\alpha = 0.1$–0.4 yield 56.6–58.6, whereas the more permissive $\alpha = 0.05$ reduces accuracy to 50.0 (Appendix B.5). Gains persist across the tested strengths but weaken at the most permissive threshold, supporting confidence-constrained execution. A mask-only ablation would be needed to separate the contrast signal's contribution from the support constraint.

4.4 Comparison at Matched Generation Budgets

Budget accounting. We compare EMO with SC and judge-selected BoN under top-p sampling, using three seeds (Table 4, Figure 2; full results in Appendix B.4.4). SC/BoN share a sample pool and match EMO's decoding settings, per-response token limits, and seeds; BoN also uses the same judge and tie-breaking rule. One initial generation and two context-conditioned streams per optimization round give EMO@K a budget of $N = 2K + 1$ target-model generation equivalents, matched to N independent samples for SC/BoN. The EMO pool contains only $K + 1$ candidates because the two streams jointly generate one response per round. This accounting excludes judge/rewrite calls and response-length differences; total FLOPs and latency are not matched.

Clear advantages on Llama, task-dependent gains on Qwen. At $N = 9$, EMO exceeds BoN on all three Llama-3.1-8B tasks: GPQA-D (37.4 vs. 32.2), MATH-500 (65.6 vs. 60.8), and MMLU-Pro (68.2 vs. 62.5). It also exceeds SC on all three. On Llama/MMLU-Pro, BoN already exceeds SC by 11.3 points using the same sample pool: candidate availability alone does not ensure successful

majority voting. EMO then raises both coverage and selected accuracy beyond BoN. For Qwen3-8B, EMO gains 2.5 points over BoN on MMLU-Pro, but the endpoint means are close on GPQA-D (60.3 vs. 60.4) and MATH-500 (85.0 vs. 84.9). SC is strongest on Qwen3-8B/MATH-500 (85.8). In the two near-ties on Qwen, EMO also lacks an oracle-coverage advantage (Appendix Table 13); its clearest accuracy gains coincide with expanded candidate coverage.

4.5 Iteration Dynamics and Selection

Early gains with task-dependent returns to further iterations. In the sampling experiments, the first two optimization rounds capture roughly 70–89% of the final gain in five of six model–task pairs (Appendix Table 10). Qwen3-8B/GPQA-D is the exception: $K = 2$ captures about 46%, and accuracy continues to improve through $K = 4$. These trajectories motivate task-dependent iteration budgets, with additional rounds remaining useful where early gains are limited.

Candidate coverage and answer selection impose different limits. At $N = 9$, EMO's five-candidate oracle coverage exceeds that of nine independent samples in four of six settings (Appendix Table 13). Figure 2 illustrates the largest coverage advantage, on Llama-3.1-8B/GPQA-D. EMO already reaches 35.5 accuracy at $K = 1$ ($N = 3$), exceeding BoN with ten samples (33.0). Its dashed oracle curve separates further from independent sampling, reaching 73.7 versus 60.3 at $N = 9$, despite using five rather than nine candidates. However, the growing gap between EMO's dashed and solid curves exposes a selection bottleneck: from $K = 1$ to $K = 4$, coverage rises from 49.3 to 73.7, while selected accuracy rises only from 35.5 to 37.4. At $N = 9$, selected accuracy as a fraction of coverage is 50.7% for EMO versus 53.4% for BoN: the advantage comes from broader coverage despite a lower fraction realized by selection (Appendix B.4.4). The opposite limitation appears on Llama-3.2-3B/AIME: even oracle coverage is only 3.3 and 6.7 on the two editions (Appendix Table 8), so better selection alone would yield limited gains. These regimes call for stronger ranking when correct candidates exist and better generation when coverage is low.

5 Conclusion

EMO connects textual feedback to token-level control by alternating question-specific context optimization and logits steering with frozen model weights. Its logit-space formulation distinguishes the direction induced by context updates from the strength and support of decoding interventions, with response feedback closing the loop. Across eight models and six benchmarks, EMO improves over Base in 39 of 40 aggregated settings, averaging $+9.10$ points. These gains show that scaling and reasoning distillation leave room for local adaptation, while variable M-step benefits motivate model- and task-specific interventions. At matched target-model generation budgets, EMO achieves higher oracle coverage with fewer candidates in four of six settings. The remaining selection gap highlights the need to pair feedback-guided generation with reliable verification. Together, these findings motivate inference systems that allocate computation jointly to feedback, distribution control, and answer selection, according to the bottlenecks of each model and task.

Limitations and future work. EMO relies on an external judge for feedback and selection, which can fail to recognize correct candidates. Calibrated ranking and process-level verification could convert candidate coverage into higher final accuracy. Steering also requires access to token distributions and two model streams, while the reported budget comparison excludes judge/rewrite overhead. Lightweight judges and adaptive stopping could reduce overhead; response-level reranking could extend feedback-guided adaptation to text-only interfaces, although it would replace token-level steering. These extensions require end-to-end latency and token-cost evaluation. Adaptive budgets require observable proxies for candidate coverage, since oracle correctness is unavailable at deployment. The alignment guarantee assumes successful context updates and faithful steering, so it does not ensure monotonic accuracy gains. Uncertainty-aware steering strength, selective M-step activation, and acceptance checks for context updates could improve robustness to unreliable feedback. Finally, evaluation is limited to reasoning benchmarks in two model families. Future work should test broader domains and tool-using agents, and combine EMO with retrieval or multiple optimization trajectories when correct candidates are rarely generated.

AI USE STATEMENT

In this work, we used generative AI tools for manuscript language polishing, standardization of figure and table formatting, and refinement and logical improvement of theoretical proofs. We have not used generative AI tools for generating core research ideas, designing the proposed method, conducting experimental experiments, producing experimental results, or writing key technical contributions, and human subject research, data collection, and dataset annotation tasks are not applicable to this work. Additionally, we used generative AI tools for structural optimization of manuscript paragraphs and grammatical error correction. We have reviewed all AI-assisted work. We manually examined, revised, and verified all AI-polished text, AI-formatted layouts, and AI-assisted proof contents to ensure logical consistency, technical accuracy, and factual correctness. All potential errors, unreasonable reasoning and inappropriate expressions generated by AI have been manually corrected. We take full responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work does not involve human subjects, sensitive private data, or ethical risk issues including discrimination, bias, privacy leakage and harmful application scenarios. All datasets adopted in this work are publicly available open-source resources with legal and standard usage specifications. There are no potential conflicts of interest, sponsorship issues or violations of research integrity. All research processes strictly comply with the ICLR Code of Ethics and maintain academic rigor, fairness and transparency. No ethical concerns exist for this submission.

REPRODUCIBILITY STATEMENT

We have made comprehensive efforts to ensure the reproducibility of this work. The core model architecture, theoretical assumptions, algorithm design logic and experimental settings are fully elaborated in the main text. Complete proof details of theoretical claims are supplemented in the appendix. Detailed data processing pipelines, experimental implementation steps and hyperparameter configurations are documented in the supplementary materials.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, volume 2024, pp. 21246–21263, 2024.

Mislav Balunovic, Jasper Dekoninck, Ivo Petrov, Nikola Jovanovi'c, and Martin T. Vechev. Matharena: Evaluating llms on uncontaminated math competitions. *ArXiv*, abs/2505.23281, 2025. URL https://api.semanticscholar.org/CorpusID:278996037.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024.

Karl Cobbe, Vineet Kosaraju, Mo Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *ArXiv*, abs/2110.14168, 2021. URL https://api.semanticscholar.org/CorpusID:239998651.

DeepSeek-AI. DeepSeek-R1-0528-Qwen3-8B: Model card. Hugging Face, 2025. URL https://huggingface.co/deepseek-ai/DeepSeek-R1-0528-Qwen3-8B.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yujiu Yang, Nan Duan, Weizhu Chen, et al. Critic: Large language models can self-correct with tool-interactive critiquing. In *International Conference on Learning Representations*, volume 2024, pp. 57734–57811, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Zhiyang He, Sashrika Pandey, Mariah Schrum, and Anca Dragan. Context steering: Controllable personalization at inference time. In *International Conference on Learning Representations*, volume 2025, pp. 87863–87895, 2025.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Xiaodong Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *ArXiv*, abs/2103.03874, 2021. URL https://api.semanticscholar.org/CorpusID:232134851.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Senkang Hu, Xudong Han, Jinqi Jiang, Yihang Tao, Zihan Fang, Yong Dai, Sam Kwong, and Yuguang Fang. Distribution-aligned decoding for efficient llm task adaptation. *Advances in Neural Information Processing Systems*, 38:56153–56188, 2026.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori B Hashimoto, Luke Zettlemoyer, and Mike Lewis. Contrastive decoding: Open-ended text generation as optimization. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 12286–12312, 2023.

Yibo Li, Zijie Lin, Ailin Deng, Xuan Zhang, Yufei He, Shuo Ji, Tri Cao, and Bryan Hooi. Just-in-time reinforcement learning: Continual learning in llm agents without gradient updates. *arXiv preprint arXiv:2601.18510*, 2026.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594, 2023.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023a.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, 2023b.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. Gpqa: A graduate-level google-proof q&a benchmark. *ArXiv*, abs/2311.12022, 2023. URL https://api.semanticscholar.org/CorpusID:265295009.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

Charlie Snell, Dan Klein, and Ruiqi Zhong. Learning by distilling context. *arXiv preprint arXiv:2209.15189*, 2022.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Ulisse Mini, and Monte MacDiarmid. Activation addition: Steering language models without optimization. 2024.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max W.F. Ku, Kai Wang, Alex Zhuang, Rongqi "Richard" Fan, Xiang Yue, and Wenhu Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *ArXiv*, abs/2406.01574, 2024. URL https://api.semanticscholar.org/CorpusID:270210486.

Wei Xia and Zhi-Hong Deng. Sda: Steering-driven distribution alignment for open llms without fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 34025–34033, 2026.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *International Conference on Learning Representations*, volume 2024, pp. 12028–12068, 2024.

Haoran Ye, Xuning He, Vincent Arak, Haonan Dong, and Guojie Song. Meta context engineering via agentic skill evolution. *arXiv preprint arXiv:2601.21557*, 2026.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *Advances in Neural Information Processing Systems*, 38:113222–113244, 2026.

Mert Yuksekgonul, Thongchareon Preeyaporn, James Boen, Sheng Liu, Zhiyuan Shen, Josephine Alvarez, and James Zou. TextGrad: Automatic "differentiation" via text. *arXiv preprint arXiv:2406.07496*, 2025.

Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, et al. A comprehensive study of knowledge editing for large language models. *arXiv preprint arXiv:2401.01286*, 2024.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, et al. Agentic context engineering: Evolving contexts for self-improving language models. In *International Conference on Learning Representations*, volume 2026, pp. 86069–86100, 2026.

Yifan Zhang, Yifan Zhou, Shijie Yang, et al. Agentic context engineering: Evolving contexts for self-improving language models. *arXiv preprint arXiv:2510.04618*, 2025.

A Proofs and Method Supplementary Details

A.1 Proof of the Unified Objective

For each question with reference response $y^\star$, define $Q(y \mid q) = \mathbf{1}\{y = y^\star\}$. Then

$$\mathrm{KL}(Q\|P_\xi) = \sum_{y:Q(y|q)>0} Q(y \mid q) \log \frac{Q(y \mid q)}{P_\xi(y \mid q)} = -\log P_\xi(y^\star \mid q). \tag{8}$$

Averaging over the empirical question distribution proves Lemma 1. The equality holds independently of which component of $\xi = (\boldsymbol{\theta}, C)$ is varied. For a general fixed target, cross-entropy equals $\mathrm{KL}(Q\|P_\xi) + H(Q)$, and the entropy term is independent of either optimization variable. Thus the minimizers of the two losses agree within each chosen feasible set; the parameter and context feasible sets need not have the same minimizer.

In the test-time setting, take the question distribution to be concentrated on the current question. The target objective remains a theoretical reference: the gold response is supplied to neither the judge nor the textual optimizer. The ideal context optimizer is defined operationally as producing a context with smaller target loss whenever an improving update is assumed to exist. No differentiable structure on the space of strings is needed for that condition, and the objective identity does not assert that arbitrary textual feedback exactly differentiates NLL.

A.2 Proof of the Unified Steering Form

Exact realization of either optimization update. At a fixed history, let $\mathbf{z} = \mathbf{z}_\xi(h)$ and $\mathbf{z}^+ = \mathbf{z}_{\xi^+}(h)$ be the before/after logits. Defining $\Delta\mathbf{z} = \mathbf{z}^+ - \mathbf{z}$ immediately gives $\mathrm{Softmax}(\mathbf{z} + \Delta\mathbf{z}) = P_{\xi^+}(\cdot \mid h)$. Equivalently, the log-ratio $\log(P_{\xi^+}/P_\xi)$ differs from $\Delta\mathbf{z}$ only by the scalar difference of log-normalizers. Softmax ignores this token-independent constant. For parameter optimization, the difference is $f_{\boldsymbol{\phi}}(C, h) - f_{\boldsymbol{\theta}}(C, h)$; for context optimization it is $f_{\boldsymbol{\theta}}(C', h) - f_{\boldsymbol{\theta}}(C, h)$. Applying the equality at every history and multiplying the conditional probabilities reproduces the optimized autoregressive distribution. For multiple optimization steps, the logit differences telescope at each fixed history, so the same statement applies to the entire resulting output change.

The ideal direct-logit gradient. For $P = \mathrm{Softmax}(\mathbf{z})$ and a fixed conditional target Q,

$$\nabla_P \mathrm{KL}(Q\|P) = -Q/P, \tag{9}$$

$$-\nabla_{\mathbf{z}} \mathrm{KL}(Q\|P) = \mathbf{J}_P(Q/P) = Q - P\sum_v Q(v) = Q - P. \tag{10}$$

The division is componentwise; finite logits ensure strictly positive P. The chain rule uses $\mathbf{J}_P^\top = \mathbf{J}_P$. Softmax ensures that the updated probabilities remain normalized. For $\boldsymbol{\delta}^* = Q - P$,

$$\frac{d}{d\eta}\mathrm{KL}(Q\|\mathrm{Softmax}(\mathbf{z} + \eta\boldsymbol{\delta}^*))\Big|_{\eta=0} = \langle P - Q, \boldsymbol{\delta}^*\rangle = -\|\boldsymbol{\delta}^*\|_2^2. \tag{11}$$

When $Q \neq P$, differentiability gives strict improvement for all sufficiently small positive η. When $Q = P$, the direction is zero and no strict improvement is claimed. For a point-mass target, the reference token receives $1 - P(y^\star)$ and other tokens receive $-P(v)$.

Which equivalence this establishes. The additive realization is exact for the output of either optimization process. The ideal gradient formula concerns direct optimization over logits. For a differentiable parameter logit map with Jacobian $\mathbf{G} = \partial\mathbf{z}/\partial\boldsymbol{\theta}$, a parameter gradient step instead induces

$$\mathbf{z}_{\boldsymbol{\theta}^+} - \mathbf{z}_{\boldsymbol{\theta}} = -\eta\mathbf{G}\mathbf{G}^\top(P - Q) + O(\eta^2), \tag{12}$$

assuming the map is twice continuously differentiable near the update. This follows by applying the chain rule to the parameter loss and Taylor-expanding the updated logits. For a training loss over multiple examples and histories, the analogous statement uses stacked logits, their Jacobian, and the full weighted loss gradient. Textual updates induce finite differences without requiring a string derivative. These observations establish a shared logit-space representation without identifying different optimizers' actual directions with $Q - P$.

Connection to reinforcement learning. At a fixed history, let P be a positive reference policy, A a fixed token advantage, and $\beta > 0$. For the unrestricted policy problem $\max_\pi\{\langle \pi, A\rangle - \beta^{-1}\mathrm{KL}(\pi\|P)\}$, define $Z = \sum_v P(v)e^{\beta A(v)}$ and $\pi^*(v) = P(v)e^{\beta A(v)}/Z$. The objective equals $\beta^{-1}\log Z - \beta^{-1}\mathrm{KL}(\pi\|\pi^*)$, so its unique maximizer is π^*. It therefore has the additive form $\pi^* = \mathrm{Softmax}(\mathbf{z} + \beta A)$. This gives the policy-improvement connection under the stated fixed-history objective; it does not identify arbitrary parameterized RL updates with direct target-KL gradient descent.

A.3 From the Context Contrast to the Steering Vector

The construction follows the probability-gradient-to-logit chain rule in SVDecode (Hu et al., 2026). Hold P_{base} fixed and differentiate at $P_{\mathrm{opt}} = \mathrm{Softmax}(\mathbf{z}^{\mathrm{opt}})$:

$$\nabla_{P_{\mathrm{opt}}}\mathrm{KL}(P_{\mathrm{opt}}\|P_{\mathrm{base}}) = \log(P_{\mathrm{opt}}/P_{\mathrm{base}}) + \mathbf{1}, \tag{13}$$

$$\nabla_{\mathbf{z}^{\mathrm{opt}}}\mathrm{KL}(P_{\mathrm{opt}}\|P_{\mathrm{base}}) = \mathbf{J}_{P_{\mathrm{opt}}}\log(P_{\mathrm{opt}}/P_{\mathrm{base}}). \tag{14}$$

The constant term vanishes because $\mathbf{J}_{P_{\mathrm{opt}}}\mathbf{1} = 0$. This proves Eq. (4); the equality defines the implemented vector, while its task-alignment interpretation remains conditional. Writing $r(v) = \log P_{\mathrm{opt}}(v) - \log P_{\mathrm{base}}(v)$ gives the centered expression

$$\delta_t(v) = P_{\mathrm{opt}}(v)\left(r(v) - \sum_w P_{\mathrm{opt}}(w)r(w)\right). \tag{15}$$

Thus a positive correction requires a log-ratio above its optimized-distribution average. The condition $P_{\mathrm{opt}}(v) > P_{\mathrm{base}}(v)$ alone is insufficient. The expression is evaluated in linear vocabulary time using log-softmax values, without forming a dense Jacobian or backpropagating through model parameters.

For the unmasked path $P(u) = \mathrm{Softmax}(\mathbf{z}^{\mathrm{opt}} + u\boldsymbol{\delta}_t)$,

$$\frac{d}{du}\mathrm{KL}(P(u)\|P_{\mathrm{base}})\bigg|_{u=0} = \|\boldsymbol{\delta}_t\|_2^2. \tag{16}$$

This explains the positive sign for local extrapolation in Eq. (4). The negative-gradient convention in SVDecode would give the opposite local change for positive strength. Neither sign determines target improvement without a target-alignment condition. In particular, the log-ratio in Eq. (4) is an observable contrast, not an identity or a generally consistent estimate of Q/P_{opt} in Lemma 2. After masking, the direction is $\hat{\boldsymbol{\delta}}_t$ and all target-alignment reasoning must use this actual vector.

Complementarity and sequential dependence. The steering operator supplies an execution rule, whereas feedback and context optimization supply task information. For any fixed direction $\mathbf{d}$, the derivative of target KL at zero is $\langle P - Q, \mathbf{d}\rangle$, so its usefulness depends on the target. EMO obtains its direction only after C_{k+1} has been estimated; the pair (C_k, C_{k+1}) then determines the contrast at each prefix. The resulting steered response drives the next context update. This is the ordered, complementary relationship underlying the E/M terminology. Classical EM monotonicity would require a specified latent-variable objective and corresponding expectation and maximization steps, which are not supplied by this structural correspondence alone.

A.4 Proof of Conditional One-Round Improvement

Fix a question and the two contexts, and abbreviate the sequence target by Q. Use finite vocabulary and maximum horizon T, finite logits at nonterminal histories, and a fixed threshold α. After EOS, pad with shared deterministic EOS transitions and zero directions; their KL contributions are zero. The distributions below are full softmax distributions before temperature scaling or decoder truncation. Let $\widetilde{P}_\mu$ denote the steered sequence distribution, so $\widetilde{P}_0 = P_{\boldsymbol{\theta},C_{k+1}}$. For a fixed history, $\hat{\boldsymbol{\delta}}_t(h)$ depends only on the contexts and α, not on μ.

The two conditions. The oracle E-step condition is

$$\mathrm{KL}(Q\|P_{\boldsymbol{\theta},C_{k+1}}) < \mathrm{KL}(Q\|P_{\boldsymbol{\theta},C_k}). \tag{17}$$

Faithful steering is made precise by requiring positive aggregate alignment of the actual masked direction with the ideal target descent direction:

$$\sum_{t=1}^{T} \mathbb{E}_{Y_{<t}\sim Q} \left\langle Q_t(\cdot \mid h) - P_{\mathrm{opt}}(\cdot \mid h), \hat{\boldsymbol{\delta}}_t(h) \right\rangle > 0. \tag{18}$$

This condition includes the threshold and finite penalty from Eq. (6). It permits some histories to have unfavorable directions, provided their target-weighted aggregate is favorable.

Conditional derivative and sequence composition. For $\widetilde{P}_\mu(\cdot \mid h) = \mathrm{Softmax}(\mathbf{z}_t^{\mathrm{opt}} + \mu\hat{\boldsymbol{\delta}}_t)$, differentiation gives

$$\frac{d}{d\mu}\mathrm{KL}(Q_t \| \widetilde{P}_\mu(\cdot \mid h))\Big|_{\mu=0} = -\langle Q_t - P_{\mathrm{opt}}, \hat{\boldsymbol{\delta}}_t \rangle. \tag{19}$$

Expanding the sequence log-ratio into conditional log-ratios gives the KL chain rule:

$$\mathrm{KL}(Q\|\widetilde{P}_\mu) = \sum_{t=1}^{T} \mathbb{E}_{Y_{<t}\sim Q} \mathrm{KL}\Big(Q_t(\cdot \mid h)\|\widetilde{P}_\mu(\cdot \mid h)\Big). \tag{20}$$

The vocabulary and horizon are finite, so differentiation may be interchanged with these finite sums. The derivative of sequence KL at $\mu = 0$ is the negative of Eq. (18), hence strictly negative. By differentiability there is $\varepsilon > 0$ such that every $0 < \mu < \varepsilon$ gives

$$\mathrm{KL}(Q\|\widetilde{P}_\mu) < \mathrm{KL}(Q\|P_{\boldsymbol{\theta},C_{k+1}}) < \mathrm{KL}(Q\|P_{\boldsymbol{\theta},C_k}), \tag{21}$$

where the second inequality is Eq. (17). This proves Theorem 1.

Role of the confidence constraint. The alignment contribution at a history is

$$\langle Q_t - P_{\mathrm{opt}}, \hat{\boldsymbol{\delta}}_t \rangle = \sum_{v:\mathbb{I}_t(v)=1} (Q_t(v) - P_{\mathrm{opt}}(v))\delta_t(v) + \lambda \sum_{v:\mathbb{I}_t(v)=0} (Q_t(v) - P_{\mathrm{opt}}(v)). \tag{22}$$

Changing α changes both sets and can therefore change the sign of this contribution. The claim applies to the masked direction at the chosen α, not merely the raw contrast. The sufficient interval for μ depends on these directions and the unknown target; the proof does not certify the empirical default or yield a test-time tuning rule. The oracle E-step condition does not by itself imply Eq. (18). Across rounds, the comparison distribution above is the preceding unsteered context distribution, not the preceding round's steered distribution. These distinctions prevent a conditional one-round result from being read as unconditional iterative convergence or an accuracy guarantee.

A.5 Non-Regression under Judge Selection

Store the score $a_k = \mathcal{J}(q, R_k)$ once for each response, and select the earliest candidate attaining the largest score. Then the selected score is at least every stored score, including a_0. Adding candidates can only increase or preserve this maximum. This proves the non-regression observation when Score denotes the stored judge score. It does not establish non-regression of rule-based correctness, which can disagree with judge ranking.

A.6 Implementation Conventions

The steering strength defaults to $\mu = 2.4$ and the threshold to $\alpha = 0.1$; the finite penalty is fixed at $\lambda = -100$. An excluded token therefore receives an additive logit change $\mu\lambda$; this is distinct from overwriting its final logit with -100. The same prefix is appended to separate KV caches for C_k and C_{k+1}. Contrast and masking are computed before any sampling temperature or top-p truncation. No context is carried from one test question to the next. The pattern filter restricts answer-bearing rewrites but cannot guarantee that all semantic hints are excluded; the judge's feedback remains an external source of task information.

With $K = 4$ refinement rounds, the initial response and four steered responses form five candidates. Each M-step uses two target-model context streams, giving nine generation equivalents including initialization. This counts streams rather than exact FLOPs or latency and excludes external judge, backward, and rewrite cost as well as differences in context and response lengths.

B EXPERIMENTS SUPPLEMENTARY DETAILS

B.1 DATASETS

We evaluate on six benchmarks spanning free-form mathematical reasoning and multiple-choice scientific knowledge, summarized in Table 5. All datasets are preprocessed into a unified schema $\{\texttt{id}, \texttt{task}, \texttt{question}, \texttt{choices}, \texttt{answer}\}$, where $\texttt{task} \in \{\texttt{math}, \texttt{math_boxed}, \texttt{mc_boxed}\}$ determines both the prompt template and the rule-based scoring rule (Appendix B.3).

Table 5: **Benchmarks.** Task format, size, and answer form. MC = multiple choice. MMLU-Pro is our stratified 560-question subset (14 categories $\times$ 40, sampling seed 42).

Benchmark	n	Task type	Answer form	Source
AIME-2025	30	math_boxed	integer	(2025)
AIME-2026	30	math_boxed	integer	(2025)
GPQA-Diamond	198	MC (A–D)	option letter	(2023)
GSM8K	1,319	math	`#### number`	(2021)
MATH-500	500	math_boxed	LaTeX / numeric	(2021)
MMLU-Pro (sampled)	560	MC (A–J)	option letter	(2024)

AIME 2025/2026. The American Invitational Mathematics Examination is a competition-mathematics benchmark with 30 problems per edition; answers are integers in $[0, 999]$. We use the 2025 and 2026 editions as separate test sets (30 problems each) and merge them in the main table for noise reduction ($n = 30$ per year yields a granularity of 3.3 points).

GPQA-Diamond. The Diamond subset of the Graduate-Level Google-Proof Q&A benchmark consists of 198 multiple-choice science questions written and validated by domain experts (physics, chemistry, biology). Each question has four options; we score the letter inside `boxed{}` (falling back to an `Answer: X` pattern).

GSM8K. Grade School Math 8.5K contains 1,319 grade-school word problems in its test split. Following the original format, models must write the final numerical answer on the last line as `#### <number>`; scoring is exact numeric match after stripping commas and decimal points. GSM8K is the most saturated benchmark in our suite (three models exceed 89% baseline accuracy), which bounds the observable headroom.

MATH-500. The 500-problem subset of the MATH competition dataset introduced by the Lightman et al. split and popularized as the MATH-500 evaluation set, covering seven subjects with LaTeX-form answers (e.g., $\frac{1}{2}$, $-\sqrt{3}$, $5r^5$). This benchmark stresses symbolic answer matching; our scoring pipeline handles numeric tolerance, LaTeX normalization, and SymPy-based symbolic equivalence (Appendix B.3).

MMLU-Pro (stratified subset). The full MMLU-Pro test set contains 12,032 questions across 14 subject categories with up to 10 options each. For compute efficiency we evaluate on a stratified subset of 560 questions: within each of the 14 categories we sample 40 questions uniformly at random with a fixed seed (`seed=42`), preserving the category distribution of the full set. The sampling procedure is implemented in `datasets/process_mmlu_pro.py` and is exactly reproducible:

```
rng = random.Random(42)
sampled = [rng.sample(cat_items[c], 40) for c in categories]
```

Prompts are constructed from the question stem and the full option block, with the output format `Answer: $LETTER$` and a step-by-step instruction. The three-task sampling main table (top-p, 3 seeds) uses GPQA-Diamond, MATH-500, and MMLU-Pro to cover science MC, math free-form, and general-knowledge MC; we exclude AIME ($n = 30$, high variance) and GSM8K (saturated) from that analysis.

B.2 Model Selection

The eight evaluated models are listed in Table 6, organized around four analysis axes with Qwen3-8B as the anchor. All models are loaded from their official Hugging Face weights with default configuration (`dtype=auto`).

Table 6: **Evaluated models.** "Thinking" marks models with an explicit long chain-of-thought mode. The selection spans scale (3B–32B), three Qwen generations, two families, and thinking/non-thinking modes.

Model	HF checkpoint	Axis probed	Thinking
Llama-3.2-3B-Instruct	`meta-llama/Llama-3.2-3B-Instruct`	scale (low)	✗
Llama-3.1-8B-Instruct	`meta-llama/Llama-3.1-8B-Instruct`	family, non-thinking	✗
Qwen2.5-7B-Instruct	`Qwen/Qwen2.5-7B-Instruct`	generation (prev), non-thinking	✗
Qwen3-8B	`Qwen/Qwen3-8B`	anchor	✓
Qwen3.5-9B	`Qwen/Qwen3.5-9B`	generation (newest)	✓
Qwen3-14B	`Qwen/Qwen3-14B`	scale (mid)	✓
Qwen3-32B	`Qwen/Qwen3-32B`	scale (high)	✓
DeepSeek-R1-0528- Qwen3-8B	`deepseek-ai/DeepSeek-R1-0528-Qwen3-8B`	reasoning distillation	✓

Scale. Qwen3-8B/14B/32B and Llama-3.2-3B/Llama-3.1-8B probe whether test-time gains shrink as models grow (main-table $\overline{\Delta}$ analysis). Mean gains within Qwen3 are non-monotonic ($+9.42$, $+7.44$, and $+8.68$ points for 8B, 14B, and 32B), showing that parameter scale alone does not determine adaptation benefit. The larger gains on AIME than on near-saturated GSM8K further indicate task dependence within each scale.

Generation. Qwen2.5-7B-Instruct $\rightarrow$ Qwen3-8B $\rightarrow$ Qwen3.5-9B form a three-generation line within one family. Every generation still gains $+6$ to $+10$ points on average after test-time optimization, answering the question of whether newer base models obsolete per-question adaptation: they do not.

Family. Qwen3-8B vs. Llama-3.1-8B-Instruct compares families at matched parameter scale, while retaining differences in architecture and training. Llama-3.1-8B gains 22.4 points on MATH-500, demonstrating applicability beyond the Qwen family; the comparison does not isolate an individual training choice.

Thinking mode. Qwen3-8B natively supports a thinking mode (explicit chain of thought), whereas Qwen2.5-7B-Instruct and Llama-3.1-8B-Instruct do not. Non-thinking models gain more over Base on MATH-500 ($+9.2$/$+22.4$ vs. $+8.0$), showing that the full framework also benefits models without a native thinking mode. Because the checkpoints differ, these gains do not establish that steering substitutes for thinking; on Llama-3.1-8B, TextGrad already accounts for most of the MATH-500 improvement.

Reasoning distillation. DeepSeek-R1-0528-Qwen3-8B was obtained by post-training Qwen3-8B-Base on chain-of-thought traces distilled from DeepSeek-R1-0528 (DeepSeek-AI, 2025). EMO improves its own Base on all five task groups ($+6.02$ points on average), demonstrating that this distilled checkpoint retains scope for test-time adaptation. Its mean M-step increment over TextGrad is -0.64, although MMLU-Pro improves by 1.2 points; thus the benefits of the full adaptation loop and of steering must be distinguished. A possible explanation is that distillation changes the conditional distributions on which steering operates. In Eq. 5, both the softmax Jacobian $\mathbf{J}_{P_{\text{opt}}}$ and the context-induced displacement $\Delta\mathbf{z}_t$ depend on the checkpoint; changes to either can alter the steering vector even with fixed μ and α. Our Qwen3-8B comparison is not a controlled pre-/post-distillation pair, so

this remains a mechanistic hypothesis. Testing it would require matched checkpoints, contexts, prefixes, and decoding budgets, measuring conditional entropy, context-induced logit displacements, and reference-token probability changes. The current accuracy results do not establish reduced entropy, overconfidence, or reinforcement-learning saturation. For this model we report the v2 rerun with an extended generation budget (16,384 tokens), as its verbose thinking chains saturate the default 8,192-token cap (Appendix B.3).

B.3 Experimental Settings

Metrics. *Final accuracy* is rule-based and gold-answer-based; the judge score is never used for reporting, only as the internal optimization signal.

- **math (GSM8K)**: extract the `#### <number>` final line; exact numeric match after removing commas and decimal points.
- **math_boxed (MATH-500, AIME)**: extract the content of the final `\boxed{}`; equality is decided by a four-stage cascade—(i) numeric comparison with 10^{-6} tolerance (resolving $\frac{1}{2}$ vs. 0.5), (ii) normalized string match (LaTeX normalization: stripping `\left`/`\right`, `\tfrac`→`\frac`, spacing), (iii) SymPy symbolic equivalence via `parse_latex`+`sympify` combinations, and (iv) a non-trivial-substring fallback (shorter side ≥ 2 characters and length ratio < 3).
- **mc_boxed (GPQA, MMLU-Pro)**: the letter inside `boxed{}`, falling back to an `Answer: X` regular expression; exact letter match with gold (options A–J supported).

judge-select@k: the reported EMO accuracy looks at the first $k{+}1$ rounds (iter0..iter$k$) and selects the round with the highest judge score (ties broken toward the earlier round), reporting its rule-based correctness. The main tables use judge-select@4 over five candidates: the initial response and four optimized responses. **pass@N** is the oracle upper bound: the question counts as solved if any of the first N attempts is correct; it is reported only for reference.

Hyperparameters. All experiments use four optimization rounds after the initial generation, with one E-step sample and one M-step sample per round, and the shared initial context *"You are a helpful assistant."* Steering defaults: total strength $\mu{=}2.4$, confidence threshold $\alpha{=}0.1$, per-step steering recomputation, and penalty value -100 on masked logits. The generation budget (`max_new_tokens`) is 8,192 for GPQA/MATH-500/MMLU-Pro/GSM8K and 12,288 for AIME in the greedy main experiments, with the DeepSeek-R1 v2 rerun using 16,384 (Appendix B.2); the top-p sampling experiments use 24,576 for Qwen3-8B (thinking chains are long) and 12,288 for the Llama models. Sampling experiments use temperature 0.6, top-p 0.95, top-k 0, and seeds $\{0, 1, 2\}$ with full-chain seeding (`torch.manual_seed` at all four sampling sites); each seed also keys the response cache so that seeds do not share generations. A hyperparameter sweep on Qwen3-8B/GPQA (Appendix B.5) shows gains are insensitive to the total strength ($+5.6$ to $+7.6$ points across $\mu \in [1.6, 3.2]$) and robust for $\alpha \in [0.1, 0.4]$, while $\alpha{=}0.05$ collapses ($+1.0$) because the confidence constraint becomes too loose. Table 7 consolidates all decoding and generation settings per experiment.

Sampling experiments use seeds $\{0, 1, 2\}$ with full-chain seeding; every EM run uses four optimization rounds after the initial generation, with one E-step and one M-step sample per round and the initial context *"You are a helpful assistant."*; evaluation batch sizes (questions per forward batch) are chosen per model and task to fit GPU memory under the worst-case two-context KV budget.

Judge deployment. The judge is Qwen3.5-122B-A10B (FP8), served with vLLM (tensor parallelism 4, max context 32,768, CUDA graphs enabled). Judge and backward (gradient/optimizer) calls use two engine instances with complementary settings: *scoring* runs greedily with disk caching (identical responses must receive identical scores, for reproducibility), while *feedback and context rewriting* run at temperature 0.7 with caching disabled—otherwise cached sampling would freeze the iterative refinement and collapse the 5 rounds into 1. Both engines disable the judge's thinking mode. The judge never sees the gold answer; its prompt requires it to independently solve the problem, score 0–10, and—critically—produce feedback *without revealing or hinting at the final answer* (numbers, letters, or expressions), which keeps the E-step gradient estimate free of answer leakage.

Table 7: **Consolidated experimental settings.** Generation budget (`max_new_tokens`) is per response; the M-step runs a batch of two contexts per decoding step. Judge settings are shared across all experiments (bottom block).

Experiment	Decoding	temp	top-p / top-k	`max_new_tokens`
Greedy main (8 models)	greedy	–	– / –	8,192; 12,288 (AIME); 16,384 (DR1-v2)
top-p main (Qwen3-8B)	sampling	0.6	0.95 / 0	24,576
top-p main (Llama-8B/3B)	sampling	0.6	0.95 / 0	12,288
top-k ablation	sampling	0.6	0.95 / 20	as top-p main (seed 0 only)
BoN/SC pool	sampling	0.6	0.95 / 0	as top-p main, n=10/question
Judge (Qwen3.5-122B-A10B FP8, shared by all experiments)				
Scoring		0 (greedy)	cached; `max_tokens`=2,048; thinking off	
Feedback / rewrite		0.7	top-p 0.95; caching off; thinking off	
Server		vLLM, TP=4, `max_model_len`=32,768, CUDA graphs on		

An additional regex filter rejects any rewritten context that leaks a concrete answer, retaining the previous context in that case.

Answer-extraction fix and rescoring. An early version of the `math_boxed` scorer hard-extracted digits from non-numeric boxed answers (e.g., $7\pi \rightarrow 7$), systematically under-reporting MATH-500 accuracy. After fixing the extraction cascade, we rescored *all* affected stored responses with the corrected rule (flips occur only on MATH-500; GPQA/MMLU/GSM8K/AIME are unchanged), and all numbers in this paper use the rescored values. The rescored greedy baseline of Qwen3-8B on MATH-500 (75.0) is consistent with the independently run sampling baseline (79.6 ± 0.4 at temperature 0.6), validating the fix.

Degenerate-generation handling (Llama). Llama-3.1/3.2 occasionally enter repetitive loops, filling the entire generation budget with a short cycle. Beyond the standard exact-cycle detector (10–30 token loops), we add an information-saturation detector: once a generation exceeds 4,000 tokens, it is terminated early if the novel-token rate over each of the last three 200-token windows falls below 0.10 relative to a 400-token baseline window. On a calibration set the detector hits 97% of degenerate chains with 0% false positives on normal chains and speeds up MATH-500 by $\sim 12\times$. Degenerate responses are almost always incorrect, so early termination has negligible effect on accuracy.

Two-arm protocol. The TextGrad (no-M-step) arm shares its baseline and iter-0 responses with the EMO arm by construction (`reuse-baseline/iter0-from`), so both arms start from an identical state and diverge only from iteration 1 onward. In the sampling experiments the no-M-step arm additionally reuses the same sharding, guaranteeing a 1:1 question match across arms.

Compute. Each optimization round generates one response using two context-conditioned model streams. Including the initial generation, EMO@k uses $N = 2k + 1$ target-model generation equivalents and produces $k + 1$ candidates. Section 4.4 matches this budget to N independent samples for SC/BoN; the accounting excludes judge/rewrite calls, response-length differences, and latency. Judge throughput is not a bottleneck: with CUDA graphs, the 122B FP8 judge sustains ~ 22 concurrent scoring workers per 4-GPU instance.

Self-consistency and Best-of-N baselines. The SC/BoN comparison shares the sampling pool of the top-p experiments: for each question we draw n=10 i.i.d. samples at temperature 0.6, top-p 0.95, with the same per-model generation budget and the same seeds $\{0, 1, 2\}$ as the EMO sampling runs. **BoN@N** selects, among the first N samples, the one with the highest judge score (the same scoring engine and prompt as EMO's judge-select, ties broken toward the earlier sample) and reports its rule-based correctness. **SC@N** takes a majority vote over the extracted final answers of the first N samples (the same extraction rules as the main scoring pipeline); a sample whose answer cannot be extracted abstains, and if all N abstain the first sample's rule-based verdict is used. On a vote tie, the answer with the highest total judge score among the tied candidates is selected. **pass@N** is the oracle upper bound shared by SC and BoN (any of the first N samples correct).

B.4 Full Experimental Results

B.4.1 AIME-2025 and AIME-2026 Reported Separately

Table 8 reports AIME-2025 and AIME-2026 separately (the main paper merges the two 30-problem sets). Because each year has only 30 problems, one problem corresponds to 3.3 points; small differences should be interpreted in light of this coarse resolution. EMO more than doubles Qwen3-8B's Base accuracy on both editions ($16.7 \rightarrow 43.3$ on 2025, $23.3 \rightarrow 50.0$ on 2026), whereas the Llama models remain near zero. For Llama-3.2-3B, even pass@5 reaches only 3.3 and 6.7 on the two editions. Thus a perfect selector over these pools would still solve few questions: the principal observed limitation is candidate generation. Low baseline accuracy provides headroom but does not ensure that four feedback-guided updates can access correct solutions.

Table 8: **AIME-2025 and AIME-2026 reported separately** (accuracy %, greedy; 30 problems per year, so 1 problem = 3.3 points). pass@5 is the oracle upper bound over the 5 EMO rounds. The main-paper Table 1 reports the average of the two years.

(a) AIME-2025				
Model	Base	TextGrad	EMO	pass@5
Llama-3.2-3B	0.0	**6.7**	3.3	3.3
Llama-3.1-8B	0.0	**3.3**	3.3	3.3
Qwen2.5-7B	6.7	13.3	**23.3**	26.7
Qwen3-8B	16.7	36.7	**43.3**	**60.0**
DR1-Qwen3-8B	13.3	**20.0**	16.7	33.3
Qwen3.5-9B	26.7	26.7	**53.3**	**80.0**
Qwen3-14B	16.7	30.0	**33.3**	**53.3**
Qwen3-32B	10.0	**43.3**	**43.3**	66.7
(b) AIME-2026				
Model	Base	TextGrad	EMO	pass@5
Llama-3.2-3B	0.0	**3.3**	3.3	6.7
Llama-3.1-8B	0.0	**3.3**	**3.3**	13.3
Qwen2.5-7B	3.3	**10.0**	**10.0**	16.7
Qwen3-8B	23.3	**50.0**	**50.0**	**66.7**
DR1-Qwen3-8B	3.3	**16.7**	**16.7**	43.3
Qwen3.5-9B	23.3	26.7	**36.7**	**73.3**
Qwen3-14B	23.3	**36.7**	33.3	56.7
Qwen3-32B	26.7	43.3	**53.3**	**73.3**

B.4.2 Accuracy vs. Iteration for All Models

Figures 3–8 plot the accuracy of the judge-selected response (solid lines) and the pass@n oracle upper bound (dashed lines) against the EMO iteration n, for all 8 models on each of the six datasets. All curves are computed from the greedy main experiments, with the baseline (grey dashed line) scored under the same unified judge. Throughout this subsection, iteration $n \in \{1, \ldots, 5\}$ counts the baseline round as n=1 (i.e., iteration n corresponds to index iter$n-1$ elsewhere).

Three patterns emerge. **(i) Early convergence.** Across the 43 (model, task) pairs with a non-saturated gain ($\geq$ 2 points), iteration 2 already realizes on average about half of the final gain of EMO, and on saturated tasks (GSM8K for the Qwen3 family) the curve flattens after a single iteration. The main exceptions are the hardest tasks (AIME, GPQA-D for weak models), where gains accrue through iteration 4—consistent with headroom being unlocked late when the base policy is far from the target. **(ii) Steering benefits depend on the task beyond its answer format.** On MMLU-Pro the pass@5 of EMO exceeds that of TextGrad for all 8 models (mean gap $+9.6$ points, up to $+17.9$ for Llama-3.1-8B: 74.5 vs. 56.6), and the selected accuracy likewise wins $8/8$. On free-form mathematics (MATH-500, GSM8K), the mean pass@5 gap is at most 1.4 points, suggesting that context refinement already provides most of the observed coverage. However, GPQA-D also uses multiple-choice answers and exhibits mixed selected-accuracy effects, so answer format alone cannot account for the MMLU-Pro pattern. The results motivate testing whether the quality of

context-induced distribution shifts, rather than output format, predicts when steering helps. **(iii) Candidate coverage and selection can improve at different rates.** On Qwen3-14B/GPQA-D, EMO reaches 79.8 pass@5 against 65.2 selected accuracy. The gap shows that correct candidates are available for additional questions but are not always selected; by itself, it does not measure semantic diversity. Rising coverage across rounds establishes that later attempts solve previously uncovered questions, motivating stronger verification to realize these gains (Section 4.5).

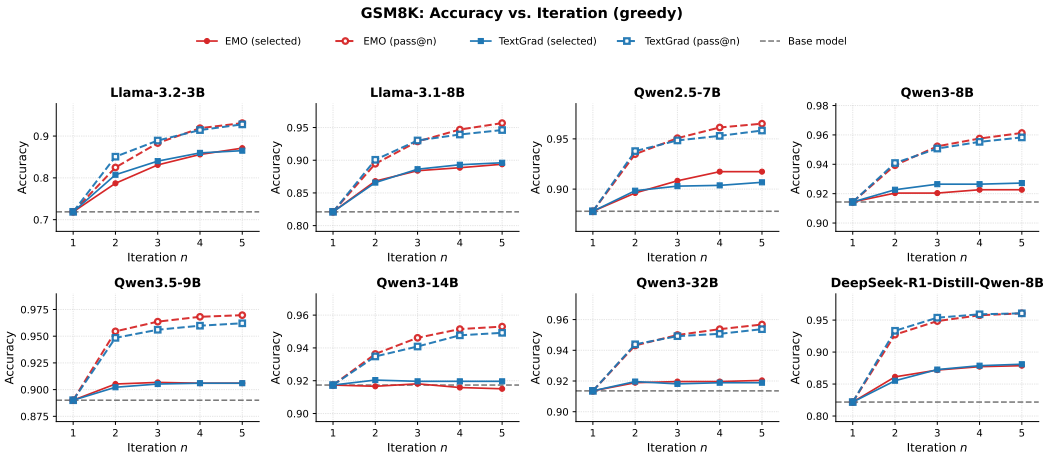


Figure 3: GSM8K: accuracy vs. EMO iteration for all models (greedy). Solid: judge-selected accuracy; dashed: pass@n (oracle); grey: base model.

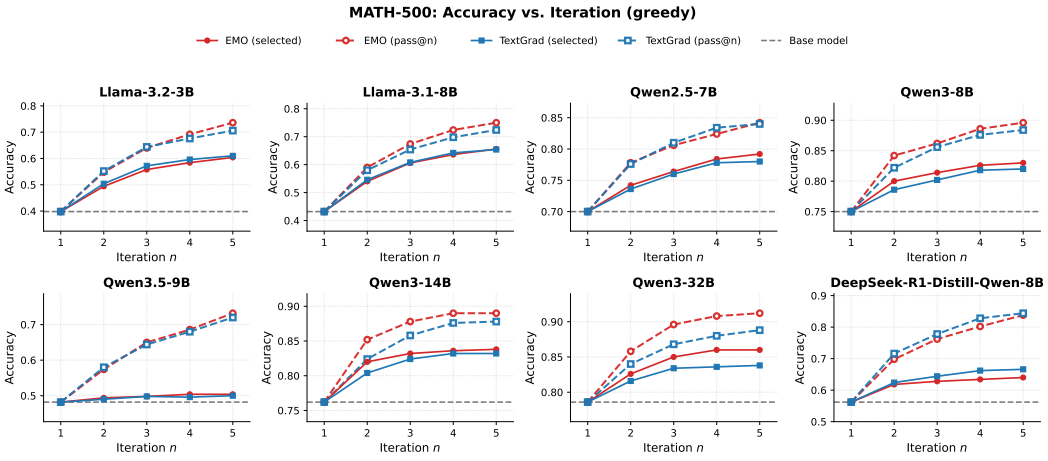


Figure 4: MATH-500: accuracy vs. EMO iteration for all models (greedy).

B.4.3 Component Ablation on Qwen3-32B

Table 9 repeats the component ablation of Table 2 on Qwen3-32B. The cross-scale comparison distinguishes three forms of component dependence. First, steering and external judging can contribute differently even within AIME. With the external judge, the M-step ties context-only optimization on AIME-2025 (43.3) but adds 10.0 points on AIME-2026 (53.3 vs. 43.3). With steering retained, external judging adds 13.3 and 6.6 points over self-judging, respectively; neither component is uniformly dominant across editions. Second, steering on GPQA-D loses 2.5 points under both judge choices (62.1 vs. 64.6 externally judged; 61.6 vs. 64.1 self-judged). This differs from Qwen3-8B, where replacing the self-judge with the external judge changes the steering increment from -3.5 to $+2.1$. Feedback quality and steering utility can interact, but external judging does not universally resolve an unfavorable steering effect. Third, with steering retained, the external-judge increment on

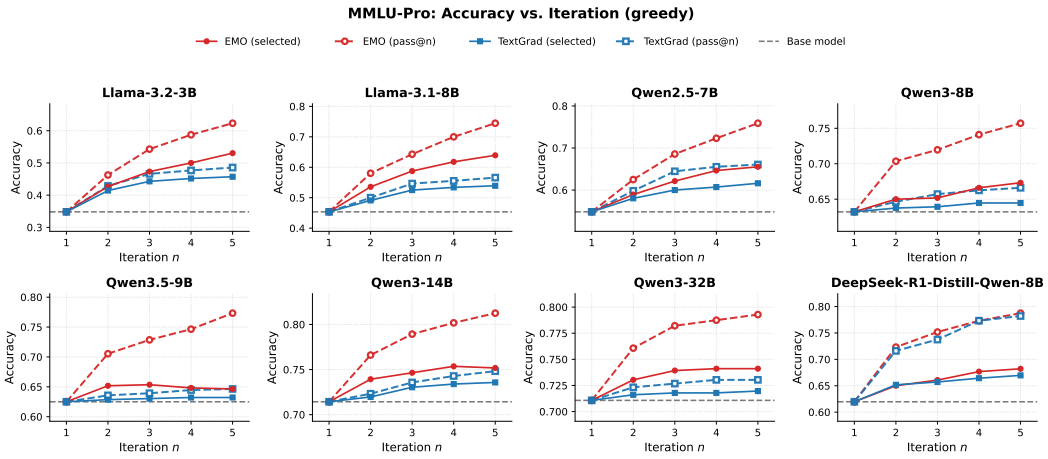


Figure 5: MMLU-Pro: accuracy vs. EMO iteration for all models (greedy).

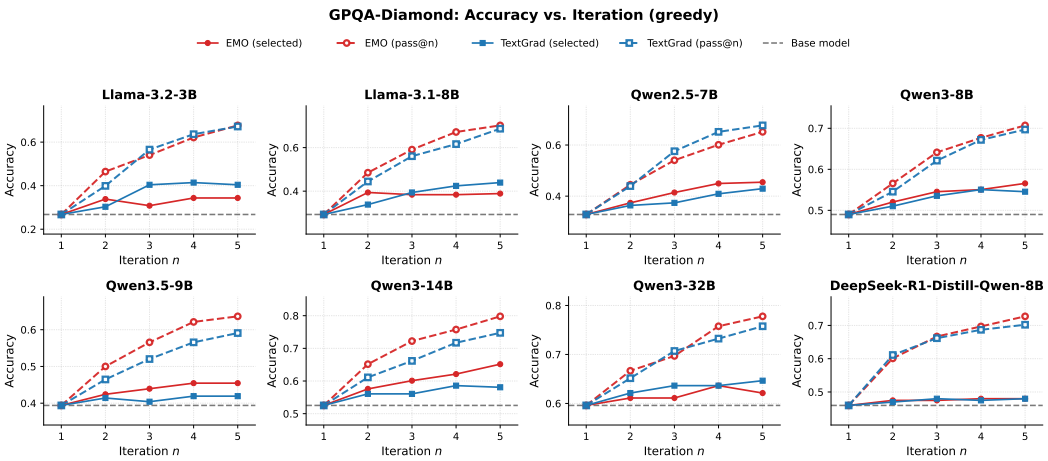


Figure 6: GPQA-Diamond: accuracy vs. EMO iteration for all models (greedy).

MMLU-Pro changes from $+2.3$ at 8B to -1.1 at 32B, where self-judging reaches 75.4 versus 74.3. This reversal motivates model- and task-dependent judge allocation, but does not establish better intrinsic self-evaluation at 32B: judge replacement changes both feedback and candidate selection, and the greedy comparison does not estimate uncertainty across runs. Occasional repetitive-generation failures under self-judging are documented in Appendix B.3.

Table 9: **Component ablation on Qwen3-32B** (accuracy %, greedy), mirroring Table 2 (Qwen3-8B). AIME rows are reported per year (30 problems each).

Dataset	Base	EMO	w/o M-step	w/o Judge	w/o Both
AIME-2025	10.0	**43.3**	**43.3**	30.0	26.7
AIME-2026	26.7	**53.3**	43.3	46.7	43.3
GPQA-D	59.6	62.1	**64.6**	61.6	64.1
GSM8K	91.4	**92.0**	91.9	91.7	91.7
MATH-500	78.6	**86.0**	83.8	83.2	82.6
MMLU-Pro	71.4	74.3	72.3	**75.4**	74.8

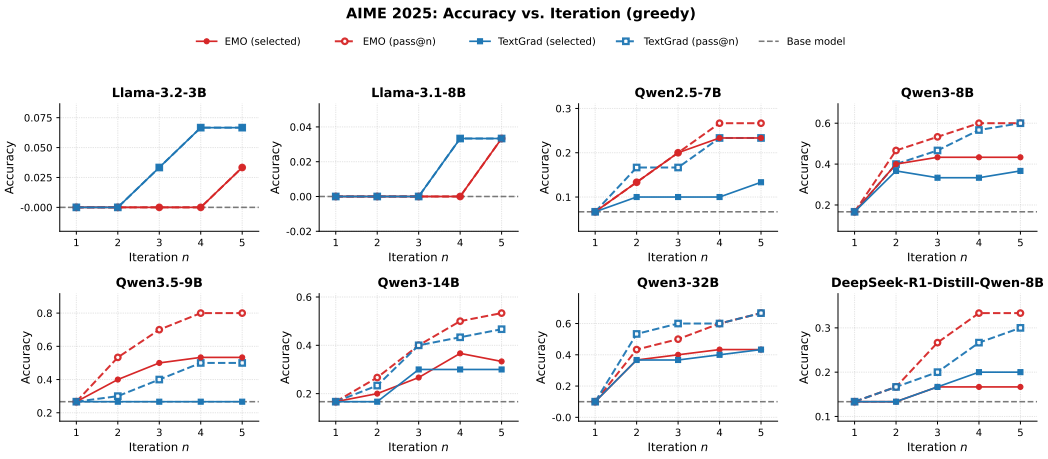


Figure 7: AIME-2025: accuracy vs. EMO iteration for all models (greedy).

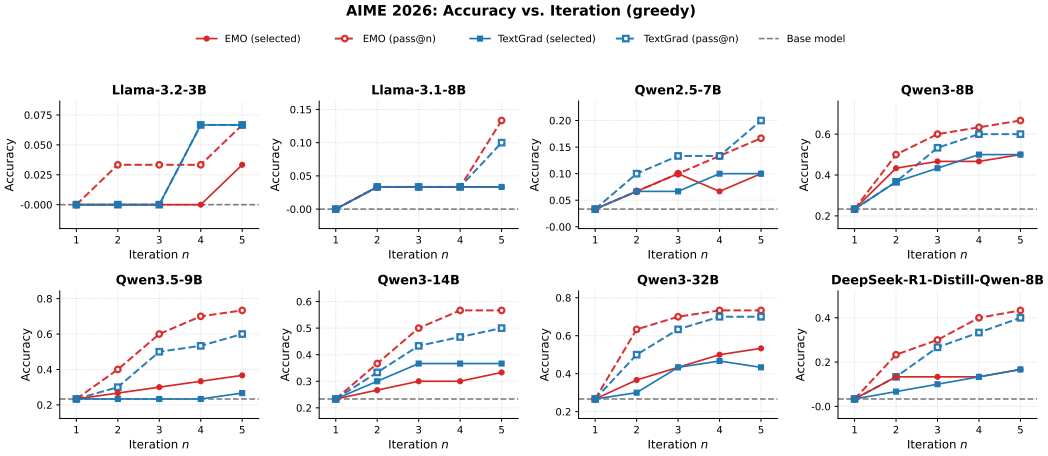


Figure 8: AIME-2026: accuracy vs. EMO iteration for all models (greedy).

B.4.4 Full Compute-Matched Results

Tables 10–13 report the full sampling results for two models and three tasks (mean±std over three seeds, temperature 0.6, top-$p = 0.95$). EMO uses $N = 2k + 1$ target-model generation equivalents; SC and BoN are reported at every budget $N = 1, \ldots, 10$. The candidate pools contain $k + 1$ responses for EMO and N for SC/BoN.

Selected accuracy. At $k = 4$ ($N = 9$), EMO exceeds BoN in mean accuracy on five of six pairs; Qwen3-8B/GPQA-D is the exception (60.3 vs. 60.4). The Qwen3-8B/MATH-500 means are also close (85.0 vs. 84.9), and SC is higher on this pair (85.8). The clearest EMO advantages over BoN are on the three Llama-3.1-8B tasks (+5.2, +4.8, and +5.7 points on GPQA-D, MATH-500, and MMLU-Pro) and Qwen3-8B/MMLU-Pro (+2.5). SC and BoN trajectories need not be monotone: for example, Qwen3-8B/GPQA-D BoN accuracy decreases from 60.6 at $N = 8$ to 59.8 at $N = 10$. Likewise, EMO's mean selected accuracy increases with each round in five pairs, while Llama-3.1-8B/GPQA-D decreases from 37.5 at $k = 3$ to 37.4 at $k = 4$.

Oracle coverage. At $N = 9$, EMO's five-candidate coverage exceeds the nine-sample SC/BoN coverage in four of six pairs. The largest gap is on Llama-3.1-8B/GPQA-D (73.7 vs. 60.3); the two exceptions are Qwen3-8B/GPQA-D (79.1 vs. 80.0) and Qwen3-8B/MATH-500 (90.4 vs. 90.9). Coverage can continue to improve while selected accuracy changes little, as on Llama-3.1-

8B/GPQA-D, indicating that candidate availability and judge selection place different limits on the final result.

How coverage translates into selected accuracy. Because a correct selected response must belong to a pool containing a correct response, selected accuracy divided by oracle coverage measures the fraction of covered questions on which selection succeeds. We compute this descriptive ratio from the reported three-seed means; with equal evaluation-set sizes, it corresponds to pooling question–seed outcomes, rather than averaging per-seed ratios. At $N = 9$ on Llama-3.1-8B/GPQA-D, the ratio is $37.4/73.7 \approx 50.7\%$ for EMO and $32.2/60.3 \approx 53.4\%$ for BoN. EMO's accuracy advantage therefore coexists with a lower fraction of available successes realized by selection: its greater coverage is essential to the observed advantage. On Llama/MATH-500, both coverage (73.4 vs. 71.9) and this fraction (89.4% vs. 84.6%) favor EMO. These are descriptive relationships, not isolated causal effects of ranking: candidate pools and the questions they cover differ across methods. They show why a larger absolute oracle–accuracy gap need not indicate less useful generation, and why selection quality alone cannot explain all accuracy gains.

Repeated sampling and feedback address different limitations. At $N = 9$ on Llama/MMLU-Pro, SC and BoN share 76.4 oracle coverage but achieve 51.2 and 62.5 accuracy, respectively. Thus many available correct responses are not recovered by majority voting, while judge selection realizes more of the same pool's potential. EMO further raises coverage to 83.2 and accuracy to 68.2 with only five candidates, showing additional value from changing how subsequent attempts are generated. For Qwen/MATH-500, by contrast, SC is strongest (85.8), and EMO has no coverage advantage over independent sampling (90.4 vs. 90.9). This contrast suggests that feedback-guided adaptation is most useful where independent sampling and its selection rule leave recoverable errors; it need not dominate when repeated sampling already supplies effective candidates and reliable consensus.

Table 10: **EMO vs. iteration under top-p sampling** (accuracy %, mean±std over 3 seeds). iter k uses $N{=}2k{+}1$ target-model generation equivalents; *Pass* is oracle coverage over the $k+1$ candidate responses.

Model	Metric		iter (N)				
			0 (1)	1 (3)	2 (5)	3 (7)	4 (9)
Qwen3-8B	acc	GPQA-D	54.4±1.4	55.4±0.2	57.1±1.1	58.4±1.3	**60.3±2.1**
	pass		54.4±1.4	64.0±0.2	70.0±1.7	75.6±1.6	**79.1±0.6**
	acc	MATH-500	79.6±0.4	83.5±0.6	84.3±0.7	84.9±0.6	**85.0±0.6**
	pass		79.6±0.4	86.9±0.6	88.9±0.2	90.0±0.2	**90.4±0.2**
	acc	MMLU-Pro	63.9±1.3	67.9±1.1	69.4±0.6	70.0±0.8	**70.9±0.7**
	pass		63.9±1.3	75.8±1.5	80.3±1.7	83.0±2.2	**84.6±1.7**
Llama-3.1-8B	acc	GPQA-D	29.6±1.3	35.5±3.6	36.5±2.5	**37.5±2.4**	37.4±0.7
	pass		29.6±1.3	49.3±2.7	62.1±1.2	70.2±1.2	**73.7±0.4**
	acc	MATH-500	45.9±1.6	55.5±1.8	59.7±1.6	63.7±1.5	**65.6±1.6**
	pass		45.9±1.6	59.3±1.7	66.1±1.7	70.5±2.1	**73.4±2.0**
	acc	MMLU-Pro	45.1±1.0	59.4±2.6	63.7±1.5	65.7±1.1	**68.2±0.8**
	pass		45.1±1.0	66.9±2.1	76.2±2.1	80.2±1.3	**83.2±1.1**

Table 11: **Self-consistency (SC@N) under top-p sampling** (accuracy %, mean±std over 3 seeds), majority vote over N independent samples.

Model	Task	SC@N, N=1, . . . , 10									
		1	2	3	4	5	6	7	8	9	10
Qwen3-8B	GPQA-D	52.4±1.4	57.4±1.0	56.4±1.9	57.6±0.8	57.6±1.6	58.1±2.2	58.1±1.9	57.6±2.7	57.7±3.0	**56.9±1.2**
	MATH-500	79.2±0.2	82.7±0.7	84.2±1.1	84.6±0.9	85.2±0.4	85.4±0.4	85.7±0.2	85.9±0.4	85.8±0.4	**86.1±0.4**
	MMLU-Pro	65.5±1.2	66.4±0.7	66.8±1.2	67.4±0.9	67.4±1.0	67.5±0.9	67.7±0.3	67.6±0.9	67.1±1.2	**67.9±0.7**
Llama-3.1-8B	GPQA-D	26.4±0.9	27.8±1.5	28.6±1.7	28.1±1.0	27.9±1.9	28.3±1.2	**29.5±1.6**	28.6±0.9	28.5±0.9	29.0±1.7
	MATH-500	45.3±1.9	51.7±1.8	53.4±1.9	54.7±1.9	55.3±1.2	55.5±1.1	56.3±0.4	55.9±0.1	56.9±0.2	**57.2±0.6**
	MMLU-Pro	45.1±0.6	**51.7±0.9**	50.4±1.5	50.8±0.5	49.7±0.4	51.4±0.5	50.7±1.0	51.1±0.5	51.2±0.8	51.5±0.7

Table 12: **Best-of-N (BoN@N) under top-p sampling** (accuracy %, mean±std over 3 seeds), judge-selected best of N independent samples.

Model	Task	BoN@N, N=1, . . . , 10									
		1	2	3	4	5	6	7	8	9	10
Qwen3-8B	GPQA-D	52.4±1.4	57.4±1.0	58.2±0.6	59.9±0.6	59.8±0.6	59.9±0.5	**60.3±0.2**	60.6±0.0	60.4±0.9	59.8±0.6
	MATH-500	79.2±0.2	82.7±0.7	84.1±0.7	84.6±0.6	84.7±0.5	84.9±0.7	85.0±0.6	85.0±0.7	84.9±0.6	**85.1±0.6**
	MMLU-Pro	65.5±1.2	66.4±0.7	67.1±1.0	67.7±0.4	68.2±0.7	68.0±0.5	68.2±0.6	68.3±0.9	68.4±0.5	**68.7±0.4**
Llama-3.1-8B	GPQA-D	26.4±0.9	27.8±1.5	28.6±1.3	29.5±1.0	29.5±0.9	30.0±0.2	30.8±0.4	31.6±0.9	32.2±0.9	**33.0±0.5**
	MATH-500	45.3±1.9	51.7±1.8	54.5±1.5	55.6±1.3	56.9±1.2	58.1±0.9	59.1±0.1	60.1±0.2	60.8±0.0	**61.3±0.2**
	MMLU-Pro	45.1±0.6	51.7±0.9	55.1±1.0	56.9±1.2	58.0±1.1	59.6±0.5	61.0±0.8	61.9±0.6	62.5±0.5	**62.9±0.8**

Table 13: **Oracle coverage at matched generation budgets** (accuracy %, mean±std over 3 seeds): at least one candidate is correct. At budget $N = 2k + 1$, SC/BoN use N independent samples and EMO uses $k + 1$ candidate responses.

Model	Task	N=3		N=5		N=9	
		SC/BoN	EMO	SC/BoN	EMO	SC/BoN	EMO
Qwen3-8B	GPQA-D	**69.4±0.6**	64.0±0.2	**74.7±1.1**	70.0±1.7	**80.0±0.2**	79.1±0.6
	MATH-500	**87.8±0.0**	86.9±0.6	**89.7±0.2**	88.9±0.2	**90.9±0.2**	90.4±0.2
	MMLU-Pro	75.2±0.8	**75.8±1.5**	78.7±0.3	**80.3±1.7**	81.9±0.5	**84.6±1.7**
Llama-3.1-8B	GPQA-D	45.1±1.3	**49.3±2.7**	51.2±0.9	**62.1±1.2**	60.3±2.7	**73.7±0.4**
	MATH-500	**60.7±1.6**	59.3±1.7	65.9±1.1	**66.1±1.7**	71.9±0.4	**73.4±2.0**
	MMLU-Pro	62.3±1.1	**66.9±2.1**	68.9±0.4	**76.2±2.1**	76.4±0.2	**83.2±1.1**

B.5 Hyperparameter Analysis

The M-step steering has two hyperparameters: the total strength μ and the confidence threshold α that masks low-confidence tokens (Section 3.2.2). Table 14 sweeps them on Qwen3-8B over GPQA-Diamond (greedy, all other settings identical to the main experiments).

Table 14: **Hyperparameter sweep on Qwen3-8B / GPQA-Diamond** (accuracy %, greedy; base = 49.0). Panel (a) sweeps the total strength μ at α=0.1; panel (b) sweeps the confidence threshold α at μ=2.4. The main-experiment default (μ=2.4, α=0.1) attains 56.6 (+7.6).

(a) Total strength μ (α=0.1)		
μ	EMO	Δ vs. base
1.6	56.6	+7.6
2.0	54.5	+5.6
2.4	56.6	+7.6
2.8	56.1	+7.1
3.2	54.5	+5.6
(b) Confidence threshold α (μ=2.4)		
α	EMO	Δ vs. base
0.05	50.0	+1.0
0.1	56.6	+7.6
0.2	**58.6**	**+9.6**
0.4	57.1	+8.1

Strength is insensitive. Across a 2× range of the total strength ($\mu \in [1.6, 3.2]$) the gain stays within 5.6–7.6 points with no monotone trend: neither the weakest nor the strongest setting is best, and no setting collapses. This indicates that the KL-gradient signal (Section 3.2.2) is a robust direction—once the update points toward the optimized context, its magnitude is of secondary importance.

The confidence constraint is what matters. The picture changes qualitatively for α. At α=0.05 the gain collapses to $+1.0$: with too permissive a mask, the steering vector acts on tokens where the optimized context expresses almost no preference, injecting noise rather than signal. At α=0.1–0.4 the gain is stable ($+7.6$ to $+9.6$, with the best value at α=0.2), i.e., the constraint only needs to separate confident tokens from noise, not to be tuned precisely. We conclude that the confidence mask, not the step size, is the load-bearing hyperparameter of the M-step, and keep the conservative default α=0.1 throughout the main experiments (where it already attains the $+7.6$ gain reported above).

B.6 Prompt Templates

This section documents the four prompts that drive the E-step (Section 3): the judge that scores responses, the backward engine that produces the textual gradient, the optimizer that rewrites the context, and the answer-leakage filter that guards the whole loop. All prompts are task-agnostic (the same templates are used for all six benchmarks); the question stem and option block are the only task-specific inputs.

B.6.1 Judge Scoring Prompt

The judge prompt (`_EVAL_SYSTEM_PROMPT` in `trainer.py`) instructs the judge model to grade a response on a 0–10 scale while remaining blind to the gold answer. Its key requirements are: (i) solve the problem *independently* before grading, never assuming the model's answer is correct; (ii) 10 means the final answer is correct, 0 means wrong or missing, with partial credit (1–9) for substantive progress on incorrect solutions; (iii) the feedback must locate the methodological error and how to fix it *without ever revealing or hinting at the final answer* (no numbers, letters, or expressions); (iv) no restating of the question or response.

```
Question: {question}\n\nResponse: {response}\n\nEvaluate
this response. Provide detailed feedback and a score.
```

The judge returns its output in the format `<FEEDBACK>...</FEEDBACK><SCORE>0-10</SCORE>`; the score drives judge-select (Appendix B.3) and the feedback becomes the input to the backward pass below.

B.6.2 Backward (Text Gradient) Prompt

The backward prompt (`BACKWARD_SYSTEM_PROMPT` in `llm_backward_prompts.py`) asks the engine to act as the "gradient" with respect to the context variable: given the current system prompt, the question, the model's response, and the judge feedback, produce *criticism of the context*—strategies, explanations, and methods to change—but never a rewritten version of the variable (that is the optimizer's job). It carries the same no-leakage and conciseness clauses as the judge prompt. The gradient is injected through a structured conversation template:

```
<LM_SYSTEM_PROMPT>{system_prompt}</LM_SYSTEM_PROMPT>
<LM_INPUT>{prompt}</LM_INPUT>
<LM_OUTPUT>{response}</LM_OUTPUT>
<FEEDBACK>{feedback}</FEEDBACK>
```

Note that the full response is embedded in the backward prompt; this is the main driver of the judge-context length budget ($\sim$14.5k tokens at peak for our longest responses).

B.6.3 Optimizer (Context Rewrite) Prompt

The optimizer prompt (`OPTIMIZER_SYSTEM_PROMPT` in `optimizer_prompts.py`) implements the TGD update: improve the context variable using the feedback, returning the improved variable wrapped in {`new_variable_start_tag`} {`improved variable`} {`new_variable_end_tag`} tags so it can be extracted and replace the old context directly, with an instruction to keep the improved variable concise. The update instruction passes the variable's role, its current value, and the gradient from the backward pass. What shapes the rewrite most is the *role description* attached to the context variable:

> *"system prompt to guide the model's behavior. A good system prompt establishes the model as a domain expert relevant to the task (e.g. 'You are an expert chemist . . . '), states the key reasoning approach and pitfalls to avoid, and specifies the required output format. Avoid generic phrases; be specific to the question's domain. Do NOT reveal or hint at the actual answer."*

This steers the E-step away from generic encouragements ("think carefully") toward substantive, domain-specific guidance, which is what the M-step steering vector can then convert into a decoding-time correction. Optional momentum (past iterations), constraints, and in-context-example sections exist in the template but are disabled in all experiments reported in this paper.

B.6.4 ANSWER-LEAKAGE FILTER

Two complementary guards keep the optimization signal free of the answer. First, the judge never receives the gold answer, and its prompt explicitly forbids the feedback from revealing or hinting at the final answer (Appendix B.6.1). Second, every context produced by the optimizer is passed through a regex filter (`_ANSWER_LEAK_PATTERNS` in `optimizer.py`) that rejects rewrites containing concrete answers—e.g. "the answer is X" patterns, contrastive "the answer is X, not Y" constructions, or specific numbers/letters—in which case the previous context is retained. The v2 filter no longer false-positives on the output-format directive `Answer: $LETTER$`: a literal `$` is only treated as numeric leakage when followed by a digit. Together these guards ensure that neither the judge feedback nor the optimized context can smuggle the gold answer into the generation path, so any accuracy gain is attributable to the optimization itself rather than to answer leakage.