SABER: STAGE-ADAPTIVE BRIDGING AND EARLY ROUTING FOR LARGE REASONING MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large reasoning models (LRMs) achieve strong performance through extended chains of thought, but their long reasoning trajectories incur substantial inference overhead. Existing early-exit methods reduce this cost by terminating reasoning once a correct answer becomes extractable, implicitly equating *answerability* with *response readiness*. In this work, we find that LRMs frequently become answerable well before they are ready to produce a high-quality response, revealing a *Solution–Response Gap* between answerability and response readiness, during which further computation consolidates the reasoning process and prepares the final response. To better exploit the information encoded within this gap, we introduce SABER, a stage-adaptive framework designed to harness the intermediate reasoning signals that bridge answerability and response readiness. SABER realizes this objective through two complementary modules: SCOUT identifies when the reasoning state becomes answerable, while BRIDGE efficiently exploits the remaining consolidation through compact recurrent latent computation. Across multiple reasoning benchmarks and different LRM backbones, SABER reduces average token usage by 22.1–49.6% while generally outperforming competing compression methods in accuracy and quality. Beyond these performance gains, further analyses reveal heterogeneous consolidation patterns associated with distinct reasoning behaviors and varying contributions to response quality. These findings provide the research community with a functional perspective on reasoning consolidation, shifting the focus beyond *when* to stop toward *what* computation remains useful in achieving response readiness.

1 INTRODUCTION

Large reasoning models (LRMs) achieve strong performance on mathematical and logical tasks through extended chains of thought (Guo et al., 2025), but their long trajectories incur substantial inference overhead (Chen et al., 2025). Early-exit methods address this cost by truncating intermediate reasoning, inserting an end-of-thinking marker such as `</think>`, and eliciting an answer from the resulting prefix (Zhao et al., 2026a; Liu & Wang, 2025). When the correct answer is already recoverable, the remaining reasoning is often treated as redundant.

However, our observations in Section 2 reveal a key limitation of this view, showing that part of the discarded reasoning remains useful for response consolidation. Specifically, even though an LRM is able to recover the correct answer, terminating reasoning at that point can still yield a response with repeated derivations, unsettled conclusions, or fragmented explanations. Continuing the reasoning can reduce these defects and improve final-response quality across dimensions such as convergence, directness, non-redundancy, and coherence. As illustrated in Figure 1, this progression separates *answerability* at t_A, the earliest boundary where the correct answer can be reliably recovered, from *response readiness* at a later boundary t_R, where the model can produce a settled, high-quality final response. Accordingly, we refer to the intervening reasoning span as the *Solution–Response Gap*.

This distinction creates a quality–efficiency trade-off: exiting at t_A saves tokens but may degrade the final response, whereas waiting until t_R retains the cost of explicitly verbalizing consolidation. Our central question is therefore: *How can the computation between answerability and response readiness be exploited efficiently without sacrificing final-response quality?*

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

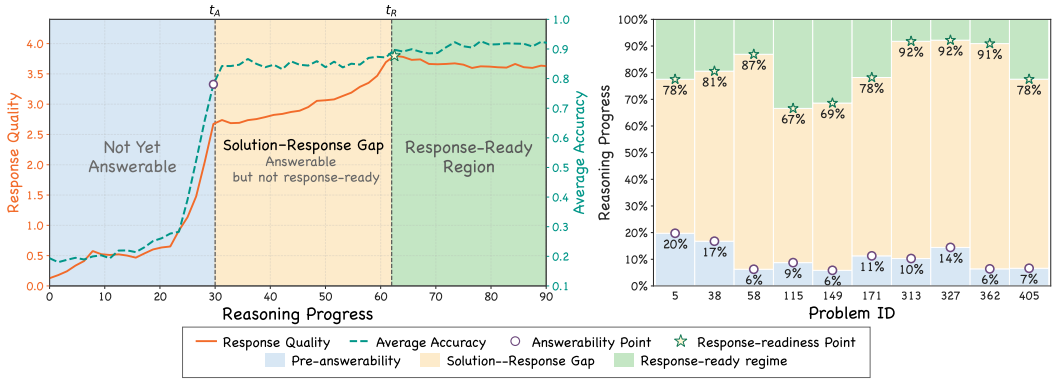

Figure 1: Answerability precedes response readiness. Left: answerability and response quality over reasoning progress. Right: example trajectories, with ○ marking the answerable point and ☆ the response-ready point. Their interval defines the *Solution–Response Gap*. Results are generated by Qwen3-4B-Thinking, with response quality evaluated by Qwen3.5-Flash.

We introduce *SABER (**S**tage-**A**daptive **B**ridging and **E**arly **R**outing)*, a stage-adaptive framework for identifying answerable reasoning states and efficiently harnessing the consolidation benefits of the Solution–Response Gap. The framework comprises two complementary modules. SCOUT determines when the current reasoning has become sufficient to produce the correct answer by leveraging the LRM's intermediate hidden states. Once SCOUT triggers, BRIDGE starts from the hidden representation of the retained reasoning prefix and recurrently generates a short sequence of latent embeddings, guiding the frozen LLM toward response readiness before final-response generation.

Across three mathematical reasoning benchmarks and a diverse set of reasoning-focused models, *SABER* reduces token usage by 22.1–49.6% while generally outperforming both training-free early-stopping and training-based compression baselines in average accuracy and response quality. Without domain-specific tuning, it also transfers effectively to scientific reasoning on GPQA-Diamond.

Beyond quantitative evaluation, we examine how consolidation varies across post-answerability reasoning states. SABER's latent representations exhibit three statistically, behaviorally, and functionally distinct modes, spanning different handoff positions and Solution–Response Gap lengths, reasoning behaviors, and response-quality gains from latent consolidation. These patterns reveal heterogeneous post-answerability computation and highlight the need to account for what useful computation remains after answerability. Our contributions are threefold:

- We formalize and empirically characterize the *Solution–Response Gap*, showing that answerability often precedes response readiness and that correctness-based early exit can discard computation that improves final-response quality.
- We develop *SABER*, a stage-adaptive framework that compresses post-answerability consolidation while keeping the LRM frozen. SCOUT identifies when explicit solution discovery is sufficient, and BRIDGE advances the state toward response readiness through compact latent computation.
- We show that *SABER* substantially reduces token usage while maintaining strong accuracy and response quality. Further analysis reveals heterogeneity in post-answerability consolidation across reasoning states, reflected in distinct latent modes and different response-quality benefits.

2 BACKGROUND AND MOTIVATION

As discussed further in Appendix A, existing methods reduce explicit reasoning in two main ways. Early-exit methods stop once sufficient answer evidence emerges (Zhao et al., 2026a; Liu & Wang, 2025; Yang et al., 2026b; Xu et al., 2026), while latent-reasoning methods replace the whole verbalized reasoning process with latent computation without distinguishing reasoning stages (Hao et al., 2024; Shen et al., 2025; Xu et al., 2025). Neither explicitly characterizes the transition from answerability to response readiness, leaving the effect of this separation on response quality largely unexplored. We formalize this separation through the *Solution–Response Gap*.

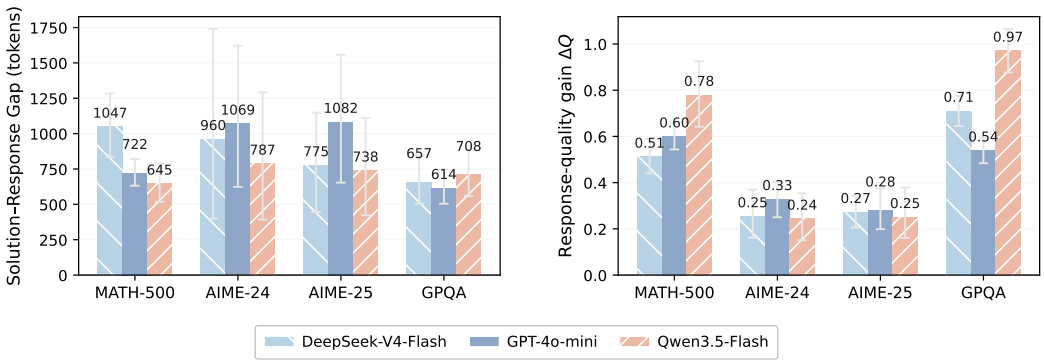

Figure 2: Left: Mean gap length Δ_{SR} between t_A and t_R. **Right:** Mean response-quality gain $\Delta Q = Q_{t_R} - Q_{t_A}$. Results are obtained with Qwen3-4B-Thinking under three rubric judges.

2.1 Definition of the Solution–Response Gap

LRMs typically separate explicit reasoning from final-response generation (Zhang et al., 2025b):

$$LRM: \ [\text{Prompt}] + \langle\text{think}\rangle + [\text{Slow Thinking}] + \langle/\text{think}\rangle + [\text{Completion}]. \tag{1}$$

We represent the explicit reasoning trajectory as $\mathcal{R} = (r_1, r_2, \ldots, r_M)$, where r_i denotes the i-th reasoning step. Following prior work (Zhang et al., 2025c), we use "`\n\n`" to separate consecutive reasoning steps and denote its i-th occurrence as boundary b_i. The accumulated reasoning prefix at b_i is $\mathcal{R}_{\le i} = (r_1, \ldots, r_i)$. We evaluate answerability and response readiness at each boundary.

Answerability. At each boundary b_i, we append the end-of-thinking marker `</think>` to the accumulated reasoning prefix $\mathcal{R}_{\le i}$ and sample K final responses. Let n_i^{correct} denote the number of responses that recover the correct answer. We define the *answerability point* t_A as

$$t_A = \min\{i : C_i \ge \tau_A\}, \qquad C_i = \frac{n_i^{\text{correct}}}{K}, \tag{2}$$

where τ_A is a reliability threshold. Thus, t_A marks the earliest point at which terminating reasoning can reliably produce the correct solution.

Response Readiness. To measure response quality, we reuse the same K final responses sampled for answerability. We evaluate their quality using a fine-grained rubric informed by FLASK and Prometheus (Ye et al., 2024; Kim et al., 2024), with process-level criteria from PRM800K and ReasonEval (Lightman et al., 2024; Xia et al., 2025). We denote the mean quality over correct responses at boundary b_i as Q_i. Let $\mathcal{I}_A$ denote the candidate boundary indices after t_A, and let $Q^\star = \max_{i \in \mathcal{I}_A} Q_i$ be the highest observed response quality. We define the *response-readiness point* t_R as

$$t_R = \min\{i \in \mathcal{I}_A : Q_i = Q^\star\}, \qquad \text{if } Q^\star > Q_{t_A}. \tag{3}$$

Finally, we define the *Solution–Response Gap* as the span between t_A and t_R, with token length Δ_{SR}. A positive Δ_{SR} indicates that answerability precedes response readiness, capturing the post-answerability evolution of response quality. Figure 1 illustrates this pervasive phenomenon.

2.2 Empirical Evidence of the Solution–Response Gap

Figure 2 shows post-answerability quality gains for Qwen3-4B-Thinking under three rubric judges. Mean gaps span 645–1,047 tokens on MATH500, 787–1,069 on AIME24, 738–1,082 on AIME25, and 614–708 on GPQA, with respective quality gains of 0.51–0.78, 0.24–0.33, 0.25–0.28, and 0.54–0.97. Together, the measurements reveal a recurring *consolidation* stage in which the solution is already recoverable but further computation improves the final response. Appendix B provides measurement and evaluation details, human validation, and robustness analyses.

Takeaway 1: [Answerability vs. Readiness] Answerability and response readiness represent distinct computational states of an LRM: the model can already know the correct answer while still requiring further computation to produce a high-quality response.

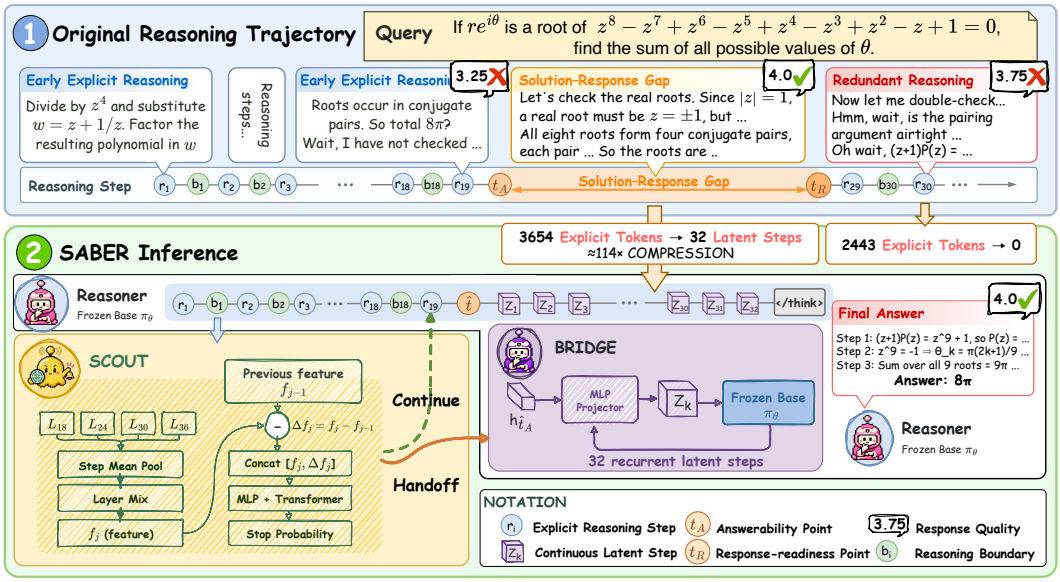

Figure 3: Overview of *SABER*. Compared with the original reasoning trajectory, *SABER* uses SCOUT to identify a handoff within the Solution–Response Gap from the LRM's hidden states, after which BRIDGE distills the explicit consolidation into compact recurrent latent computation before final-response generation.

3 METHOD

3.1 STAGE-ADAPTIVE EARLY EXIT

Cognitive studies (Miyake et al., 2004; Lidstone et al., 2010) suggest that explicit verbalization is particularly useful when a solution must still be planned or constructed, while its benefit diminishes once sufficient task-relevant information has been established. Motivated by this stage-dependent role of verbalization, we develop *SABER*, which retains explicit reasoning during solution discovery and shifts subsequent consolidation to compact latent computation.

As shown in Figure 3, the reasoning process in *SABER* unfolds autoregressively, driven by a frozen core LRM, the reasoner π_θ. Given an input x, π_θ generates an explicit reasoning trajectory $\mathcal{R} = (r_1, \ldots, r_M)$. *SABER* continuously monitors this process and adaptively determines the *handoff*, i.e., when to hand off explicit reasoning to latent consolidation. At each reasoning boundary b_i, SCOUT uses the reasoner's hidden states to represent the current reasoning state and estimates a conditional handoff probability $\widehat{h}_i$. These probabilities are accumulated as $\widehat{F}_i = 1 - \prod_{j \le i}(1 - \widehat{h}_j)$. When $\widehat{F}_i$ first reaches threshold S, we set $\widehat{t} = i$, pause explicit reasoning, and invoke BRIDGE.

Starting from the same reasoning state, BRIDGE takes the retained prefix $\mathcal{R}_{\le \widehat{t}}$ and recurrently produces K_{B} latent embeddings $z_{\widehat{t},1:K_{\mathrm{B}}}$ to advance the model toward response readiness. The frozen reasoner consumes these embeddings, processes the end-of-thinking marker $c =$ `</think>`, and generates

$$y \sim \pi_\theta\Big(\cdot \mid x, \mathcal{R}_{\le \widehat{t}}, z_{\widehat{t},1:K_{\mathrm{B}}}, c\Big). \tag{4}$$

If π_θ emits c before the handoff, it directly generates the final response without invoking BRIDGE.

3.2 SCOUT: SELECTING THE HANDOFF POINT

In this section, we describe the instantiation and training of SCOUT. SCOUT monitors the evolving reasoning state using three complementary signals: the current-step representation, its change from the previous step, and a history-aware representation derived from earlier reasoning steps.

Instantiation. Inspired by findings that intermediate and late-layer hidden states encode predictive signals of reasoning success (Afzal et al., 2025; Zhang et al., 2025a), we combine a multi-layer state

aggregator with a local MLP and a lightweight causal Transformer. At boundary b_i, let $\mathcal{S}_i$ contain the token positions in the completed step r_i. We pool residual states $h_n^{(\ell)} \in \mathbb{R}^d$ from four layers $\mathcal{L}$ at approximate relative depths $\{1/2, 2/3, 5/6, 1\}$ and mix their normalized representations:

$$\bar{h}_i^{(\ell)} = \frac{1}{|\mathcal{S}_i|} \sum_{n \in \mathcal{S}_i} h_n^{(\ell)}, \quad f_i = \sum_{\ell \in \mathcal{L}} \frac{\exp(\omega_\ell)}{\sum_{m \in \mathcal{L}} \exp(\omega_m)} \mathrm{LN}_\ell\left(\bar{h}_i^{(\ell)}\right), \quad \Delta f_i = f_i - f_{i-1}, \tag{5}$$

where LN_ℓ is layer-specific normalization, ω_ℓ are trainable mixture logits, and $\Delta f_1 = \mathbf{0}$. A projection P_η maps $u_i = [f_i; \Delta f_i]$ to $x_i = P_\eta(u_i) \in \mathbb{R}^{128}$. A two-layer MLP g_ψ provides a local logit, augmented by a two-layer, four-head causal Transformer T_η:

$$v_i = \left[T_\eta(x_{1:i} + e_{1:i}; M_i)\right]_i, \quad s_i = g_\psi(u_i) + \mathbf{a}^\top \mathrm{LN}(v_i) + b, \quad \widehat{h}_i = \sigma(s_i), \tag{6}$$

where e_i is a sinusoidal position encoding, M_i masks future steps, $\mathbf{a}, b$ are trainable readout parameters, and σ is the sigmoid.

Training Recipe. We use an *interval-based first-stop objective* to preserve solution discovery while avoiding delayed handoffs. For training trajectories $\mathcal{D}$, let $\mathcal{B}_\mathcal{R}$ denote probed boundary indices and $\mathcal{I}_\mathcal{R} = \{i \in \mathcal{B}_\mathcal{R} : t_A \le i \le t_R\}$ the target window from Section 2.1. The first-stop probability q_i and interval loss $\ell_\mathcal{R}$ are

$$q_i = \widehat{h}_i \prod_{j \in \mathcal{B}_\mathcal{R}, j < i} (1 - \widehat{h}_j), \qquad \ell_\mathcal{R} = -\log\left(\sum_{i \in \mathcal{I}_\mathcal{R}} q_i\right). \tag{7}$$

This interval objective encourages the first handoff to fall within the annotated interval $[t_A, t_R]$, preserving explicit solution discovery while reserving the remaining consolidation for BRIDGE. For trajectories without an observed gap, prefixes before t_A train SCOUT to continue explicit reasoning and avoid premature handoff. With the reasoner frozen, we train a shared SCOUT by minimizing $\mathcal{L}_{\mathrm{SCOUT}} = |\mathcal{D}|^{-1} \sum_{\mathcal{R} \in \mathcal{D}} \ell_\mathcal{R}$. Theoretical analysis and more details are provided in Appendix C.

3.3 BRIDGE: RECURRENT LATENT CONSOLIDATION

In this section, we elaborate on BRIDGE, the latent consolidation module in *SABER*. Rather than verbalizing the Solution–Response Gap, BRIDGE compresses useful consolidation information into recurrent latent computation, guiding the frozen reasoner efficiently toward response readiness.

Instantiation. BRIDGE is a two-layer MLP projector parameterized by ϕ. At handoff b_i, let $h_{i,0}$ denote the reasoner's hidden state at the final token of $\mathcal{R}_{\le i}$. For $k = 1, \ldots, K_\mathrm{B}$, it maps the preceding hidden state to an embedding consumed by the frozen reasoner:

$$z_{i,k} = \rho\, \mathrm{LN}(W_2 \cdot \mathrm{GELU}(W_1 h_{i,k-1} + \mathbf{b}_1) + \mathbf{b}_2), \quad h_{i,k} = \pi_\theta^{\mathrm{hid}}(z_{i,k} \mid x, \mathcal{R}_{\le i}, z_{i,1:k-1}), \tag{8}$$

where $\pi_\theta^{\mathrm{hid}}$ returns the hidden state at the appended latent position and $\rho > 0$ is a learnable scale. The projector is shared across steps, and the reasoner retains the full explicit and latent prefix. We use $K_\mathrm{B} = 32$ by default, then append c and generate the response as in Equation 4.

Training Recipe. For each trajectory with $t_A < t_R$, we select up to four uniformly spaced handoff boundaries within $[t_A, t_R)$ and pair each student prefix $\mathcal{R}_{\le i}$ with the teacher prefix $\mathcal{R}_{\le t_R}$. The J-token target $y^R = (y_1^R, \ldots, y_J^R)$ is the highest-scoring correct response sampled at t_R.

The student adds K_B latent embeddings, while the teacher retains explicit reasoning up to t_R. Both use the frozen reasoner, append c, and evaluate y^R under teacher forcing:

$$p_B(y_j^R) = \pi_\theta\left(y_j^R \mid x, \mathcal{R}_{\le i}, z_{i,1:K_\mathrm{B}}, c, y_{<j}^R\right), \quad p_R(y_j^R) = \pi_\theta\left(y_j^R \mid x, \mathcal{R}_{\le t_R}, c, y_{<j}^R\right). \tag{9}$$

We augment response cross-entropy with teacher-confidence weighting and cosine-distance alignment of pre-response states:

$$\mathcal{L}_{\mathrm{BRIDGE}}(\phi) = -\frac{1}{J}\sum_{j=1}^{J} \log p_B(y_j^R) - \frac{\beta}{J}\sum_{j=1}^{J} p_R(y_j^R) \log p_B(y_j^R) + \lambda\left(1 - \frac{u_B^\top u_R}{\|u_B\|_2 \|u_R\|_2}\right). \tag{10}$$

Here, u_B and u_R are the student and teacher hidden states after processing c, before any response tokens; β and λ control confidence weighting and hidden-state alignment, respectively. The reasoner remains frozen, and teacher outputs are treated as constants. Gradients pass through the student's latent recurrence and teacher-forced response prediction to update only ϕ, including ρ. Further implementation details and theoretical analysis are provided in Appendix D.

4 EXPERIMENTS

In this section, we evaluate *SABER* through four research questions:

- **RQ1:** Can *SABER* reduce token usage while preserving correctness and response quality?
- **RQ2:** Can math-trained *SABER* generalize across reasoning domains?
- **RQ3:** How do *SABER*'s components contribute to its effectiveness?
- **RQ4:** How is post-answerability reasoning organized within the Solution–Response Gap?

4.1 EXPERIMENTAL SETUP

Evaluation and Benchmarks. We evaluate *SABER* on three mathematical reasoning benchmarks: MATH500 (Lightman et al., 2024), AIME24, and AIME25. For the main mathematical benchmark evaluation, we use four random seeds with two generations per problem per seed. For cross-domain evaluation, we use the 198-question GPQA-Diamond benchmark (Rein et al., 2024).

Baselines. We compare against four reasoning-compression baselines, categorized into two groups: **(i) Training-free early stopping**, including DEER (Yang et al., 2026b), which uses answer confidence to determine when to stop reasoning, and CUSUM (Xu et al., 2026), which detects reasoning convergence through entropy-based change-point detection; and **(ii) Training-based compression**, including BCR (Yang et al., 2026a), which uses reinforcement learning to encourage shorter reasoning, and ConPress (Deng et al., 2026), which fine-tunes on concise, correct reasoning traces.

Implementation Details. We use three reasoning backbones of varying sizes: DeepSeek-R1-Distill-Qwen-1.5B and DeepSeek-R1-Distill-Qwen-7B (Guo et al., 2025), and Qwen3-4B-Thinking (Yang et al., 2025). All methods requiring training use the same 2,000 problems sampled from DeepMath-103K (He et al., 2026), with 1,600 problems for training and 400 for validation. For *SABER*, t_R and SCOUT/BRIDGE training annotations use response-quality scores from Qwen3.5-Flash.

Evaluation Metrics. We report final-answer accuracy (Acc., %), mean token usage per problem (Tok.), and mean response quality over correct responses only (Qual.). To avoid evaluator overfitting, response quality is scored by DeepSeek-V4-Flash. Since Qual. does not reflect the frequency of correct responses, we additionally report *Quality-Adjusted Accuracy* (QAA) to jointly assess correctness and response quality. Let $\mathcal{E}$ denote the set of evaluation benchmarks. QAA combines seed-averaged accuracy and conditional response quality, with equal weight across benchmarks:

$$\mathrm{QAA} = \frac{1}{|\mathcal{E}|} \sum_{d \in \mathcal{E}} \frac{\mathrm{Acc}_d}{100} \, \mathrm{Qual}_d. \tag{11}$$

For each benchmark, Acc. and Qual. are first averaged across seeds, and QAA is then computed from these aggregated benchmark-level values. Further metric details are provided in Appendix E.

4.2 MAIN RESULTS (RQ1)

As shown in Table 1, *SABER* balances token compression and task performance across mathematical reasoning benchmarks. On R1-Distill-Qwen-1.5B, it reduces token usage by 49.6% with only a 0.6-point loss in average accuracy. CUSUM saves far fewer tokens, whereas ConPress reaches similar compression at a larger accuracy cost. The benefit extends to larger backbones: on R1-Distill-Qwen-7B, token usage falls by 22.1% as average accuracy rises by 3.3 points; on Qwen3-4B-Thinking, tokens fall by 34.1% as accuracy rises by 1.7 points. Together, these results show that *SABER* shortens reasoning while preserving or improving task performance across these model scales.

Response quality adds another dimension. On MATH500 with Qwen3-4B-Thinking, *SABER* and ConPress have similar accuracies, but the mean quality of their correct responses differs sharply: 3.965 for *SABER* versus 2.082 for ConPress. This shows why accuracy alone cannot fully characterize the effects of compression on final responses. Across the 1.5B, 7B, and Qwen3-4B-Thinking backbones, *SABER* achieves the highest QAA among compression methods, scoring 1.805, 2.299, and 3.359, respectively. This indicates that its advantage extends beyond token reduction to the joint accuracy–quality measure. Together, these results demonstrate *SABER*'s ability to compress reasoning while maintaining strong correctness-adjusted response quality. Additional baseline comparisons and latency results are provided in Appendices E and G, respectively.

Table 1: Main results on mathematical reasoning benchmarks. We report accuracy (Acc.), token usage (Tok.), and response quality (Qual.) across three LRM backbones. Qual. is averaged over correct responses only, while QAA denotes quality-adjusted accuracy averaged across the three benchmarks. Best and second-best results among compression methods are shown in **bold** and underlined, respectively.

	MATH500			AIME24			AIME25			Avg.			
Model	Acc.↑	Tok.↓	Qual.↑	Acc.↑	Tok.↓	Qual.↑	Acc.↑	Tok.↓	Qual.↑	ΔAcc.↑	ΔTok.↓	ΔQual.↑	QAA↑
R1-Distill-Qwen-1.5B													
Original	84.6	6259	3.805	30.0	26720	3.417	30.4	20847	3.722	–	–	–	1.792
DEER	62.2	**2663**	**3.850**	10.0	**9826**	3.583	23.8	**8447**	3.464	-16.4	**-60.1%**	-0.016	1.192
CUSUM	84.2	5850	3.796	29.6	23067	3.389	26.7	17359	3.750	-1.5	-12.3%	-0.003	1.733
BCR	79.3	4919	3.446	30.4	16541	3.389	26.2	14221	3.656	-3.0	-30.4%	-0.151	1.574
ConPress	73.7	3288	2.031	25.4	10531	3.000	22.1	10541	3.000	-7.9	-52.5%	-0.971	0.974
Ours	**84.4**	3029	3.823	**31.7**	11106	**3.600**	**27.1**	12779	**3.875**	**-0.6**	-49.6%	**+0.118**	**1.805**
R1-Distill-Qwen-7B													
Original	92.4	4118	3.862	57.1	14280	3.647	36.2	17527	3.455	–	–	–	2.301
DEER	84.8	2184	3.755	49.2	9839	3.033	30.0	10176	3.417	-7.3	-40.0%	-0.253	1.901
CUSUM	87.2	3246	**3.834**	36.7	**4769**	**3.773**	20.4	**3485**	3.333	-13.8	**-56.0%**	**-0.008**	1.802
BCR	91.7	2757	3.609	50.0	11248	3.250	36.7	13257	3.409	-2.5	-26.2%	-0.232	2.061
ConPress	85.0	**2023**	2.065	51.2	7853	2.990	36.2	8352	2.988	-4.4	-49.4%	-0.974	1.457
Ours	**91.9**	2854	3.677	**57.1**	12876	3.359	**46.7**	13005	**3.432**	**+3.3**	-22.1%	-0.165	**2.299**
Qwen3-4B-Thinking													
Original	98.2	6634	3.890	82.1	19800	3.770	77.1	23021	3.609	–	–	–	3.232
DEER	94.5	5430	3.901	60.0	13018	3.694	60.0	13185	3.708	-14.3	-31.7%	+0.011	2.710
CUSUM	92.2	4170	3.566	77.1	17703	3.438	70.0	19021	3.012	-6.0	-21.7%	-0.418	2.682
BCR	96.4	3795	3.884	76.7	15341	3.761	76.2	19400	3.478	-2.7	-27.0%	-0.049	3.093
ConPress	96.0	**2236**	2.082	69.6	**10711**	2.991	59.2	**11871**	2.997	-10.9	**-53.5%**	-1.066	1.951
Ours	**97.8**	3842	**3.965**	**84.2**	15184	**3.781**	**80.4**	14548	**3.750**	**+1.7**	-34.1%	**+0.076**	**3.359**

4.3 Cross-Domain Generalization (RQ2)

Transfer beyond mathematical reasoning. To assess cross-domain generalization, we evaluate math-trained *SABER* on GPQA-Diamond using Qwen3-4B-Thinking, without domain-specific tuning. Table 2 shows a 16.9% reduction in token usage relative to Original, while accuracy remains near 67% and the quality of correct responses ranks highest. These results extend the observed quality–efficiency balance from mathematical to scientific reasoning.

Table 2: Out-of-domain results on GPQA-D. Best and second-best results are bold and underlined.

Method	Acc.↑	Tok.↓	Qual.↑
Original	66.16	8888.18	3.7328
DEER	**69.19**	8540.52	3.6916
CUSUM	63.13	**6996.44**	3.1900
BCR	66.16	7246.02	3.7080
SABER	66.67	7390.42	**3.7784**

Compression trade-offs on GPQA-D. The baselines expose distinct efficiency–quality trade-offs. DEER achieves the highest accuracy, but uses 15.6% more tokens than *SABER* and yields lower response quality. CUSUM provides the strongest compression, but records the lowest accuracy and quality. BCR offers the closest token budget, making it the most direct efficiency-matched comparison: *SABER* uses only 2.0% more tokens while improving both accuracy and response quality. Together, these results show that *SABER* maintains strong overall performance under domain shift, and remains competitive against diverse compression baselines. Further details appear in Appendix E.

4.4 Ablation Studies (RQ3)

SCOUT Learns When to Consolidate. We further examine where SCOUT triggers relative to the answerable boundary t_A and response-ready boundary t_R. Empirically, 84.35% of SCOUT handoffs fall within the Solution–Response Gap. Figure 4 shows their average positions, with handoffs occurring after answerability but well before response readiness on average. This placement matters: triggering too early risks truncating useful solution-forming computation, whereas triggering too late leaves little consolidation to compress. Consequently, SCOUT preserves essential explicit reasoning to maintain solution correctness, while delegating the remaining consolidation to BRIDGE. Despite large differences in absolute reasoning length, SCOUT triggers at a similar offset from t_A across benchmarks, suggesting a consistent transition from answerability to consolidation. Additional SCOUT layer ablations, sensitivity analyses, and details are provided in Appendix E.

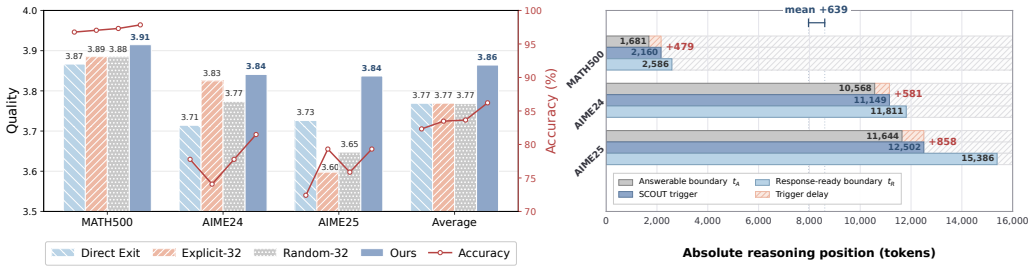

Figure 4: Left: Post-handoff comparisons on SCOUT-triggered problems, with bars showing response quality and the red line showing accuracy. **Right:** Mean token positions of t_A, SCOUT handoffs, and t_R on matched trajectories from MATH500, AIME24, and AIME25.

BRIDGE Learns How to Consolidate. Keeping SCOUT fixed, we compare SABER with three post-handoff strategies on the same triggered problems: *Direct Exit* terminates reasoning immediately; *Explicit-32* continues with up to 32 explicit reasoning tokens; and *Random-32* uses a randomly initialized projector of the same architecture. As shown in Figure 4, *SABER* achieves 86.21% average accuracy and the highest response quality. The gain over Direct Exit confirms that useful computation remains after handoff, while the advantage over Explicit-32 favors learned latent consolidation over short textual continuation. Its improvement over Random-32 further attributes this benefit to the learned latent transformation rather than additional latent computation alone. These comparisons demonstrate BRIDGE's effectiveness in consolidating useful post-handoff computation. Detailed protocols and sensitivity analyses are provided in Appendix E, with a case study in Appendix H.

4.5 UNDERSTANDING POST-ANSWERABILITY CONSOLIDATION (RQ4)

We further conduct several experiments to analyze SABER's latent organization and characterize how post-answerability consolidation is organized within the Solution–Response Gap.

Latent Modes Reflect Different Reasoning States. We mean-pool the 32 latent embeddings generated by BRIDGE and analyze the resulting representations using t-SNE and Gaussian mixture clustering. As shown on the left of Figure 5, the latent representations form three visually distinct groups. To determine whether this separation reflects meaningful differences in post-answerability reasoning rather than geometric structure alone, we quantitatively compare two trajectory-level characteristics across the groups: the SCOUT handoff position and the Solution–Response Gap length. As shown in the center of Figure 5, the three groups exhibit a clear ordered pattern: one occurs earlier with shorter gaps, another occupies an intermediate region, and the third occurs later with substantially longer gaps. This progression suggests that the latent separation captures differences in handoff timing and Solution–Response Gap length. Based on these characteristics, we refer to the three modes as *Rapid Closure*, *Intermediate Stabilization*, and *Extended Consolidation*, respectively.

Explicit Continuation Intervention. Despite their statistical distinctness, we further delve into differences in reasoning behavior among the three latent modes. Since these latent modes lack direct semantic descriptions, we use *Explicit Continuation Intervention*, which maps each uninterpretable mode back to an observable explicit reasoning trajectory. Concretely, at each SCOUT-selected handoff point, we replace BRIDGE with the original LRM and let it continue explicit reasoning naturally from the same state. We find that these continuations exhibit three recurring behavioral patterns, which can be summarized as *linear completion*, *iterative validation*, and *revision and reconciliation*. Definitions and details are provided in Appendix F.

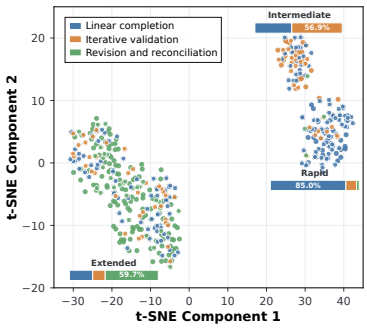

Figure 6: Behavioral composition of latent modes. Mini-bars show native behavior within each mode.

Comparing Figures 5 and 6, the three latent modes show clear correspondence with distinct native continuation behaviors. Rapid Closure is dominated by linear completion, Intermediate Stabilization by iterative validation, and Extended Consolidation by revision and reconcilia-

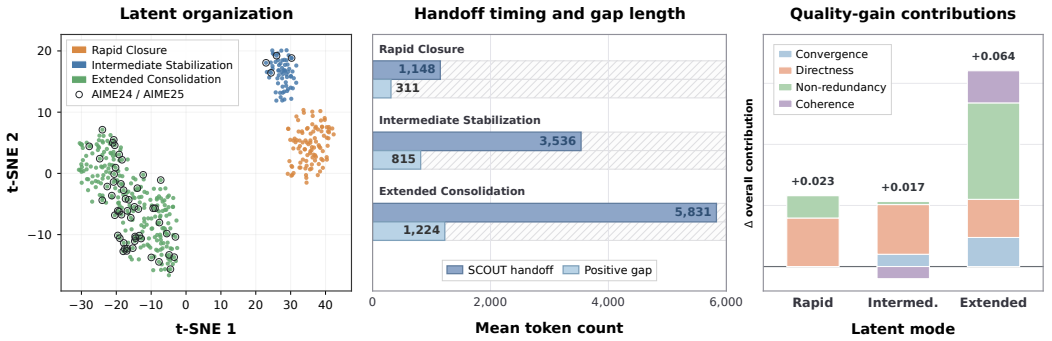

Figure 5: Three latent modes exhibit distinct handoff locations and quality-gain profiles. Left: t-SNE visualization of mean-pooled BRIDGE embeddings at SCOUT handoffs. Center: for each latent mode, the dark bar shows the mean SCOUT handoff position and the light bar shows the mean positive Solution–Response Gap length Δ_{SR}. Right: decomposition of the average response-quality improvement of *SABER* over Direct Exit into the four rubric dimensions; numbers above bars indicate the total mean quality gain for each mode.

tion. This alignment connects the implicit latent organization to observable reasoning behavior, providing a preliminary interpretation of the three modes. More importantly, it suggests that SABER's latent space is functionally organized according to how post-answerability reasoning proceeds—whether the state primarily calls for completing, validating, or revising the existing reasoning.

Consolidation Benefits Vary Across Reasoning States. To examine the functional distinctness of these modes through their benefits from latent consolidation, we compare BRIDGE with Direct Exit from the same handoff prefixes to measure the response-quality improvements attributable to consolidation in each mode. As shown on the right of Figure 5, Rapid Closure and Intermediate Stabilization exhibit highly localized improvements, with directness accounting for about 69% and 76% of their positive dimension-wise gains, respectively. In contrast, Extended Consolidation improves all four quality dimensions, with non-redundancy contributing nearly half of the total positive gain. Its overall improvement is roughly 2.8× that of Rapid Closure and 3.8× that of Intermediate Stabilization. Thus, under the same BRIDGE module and fixed latent budget, different reasoning states exhibit different response-level benefits from consolidation.

Taken together, the three analyses reveal statistically, behaviorally, and functionally distinct latent modes that emerge at different stages of post-answerability reasoning, exhibit different native continuation tendencies, and benefit differently from the same latent computation. This alignment reveals that answerability does not place all trajectories in a uniform state of response preparation. Instead, the remaining consolidation varies across reasoning states, which SABER's latent organization captures in a compact and behaviorally grounded form.

Takeaway 2: [The Solution–Response Gap Is Heterogeneous] After answerability, LRMs do not enter a uniform response-preparation state. Instead, consolidation varies across reasoning states, which SABER captures through distinct, behaviorally grounded latent modes.

5 CONCLUSION

We identified the *Solution–Response Gap*, separating when an LRM can recover the correct answer from when it is ready to produce a high-quality response. This motivates *SABER*, which identifies when reasoning enters the gap and exploits its remaining consolidation signal through compact latent computation. Across three mathematical benchmarks and three backbones, *SABER* reduces token usage by 22.1–49.6% while achieving the strongest quality-adjusted accuracy among evaluated compression methods, and further generalizes to other domains. Further analysis reveals distinct consolidation functions across reasoning states, reflected in different reasoning behaviors and dimension-specific improvements in quality. Together, these findings frame answerability not as the end of useful computation, but as a transition from solution discovery to response consolidation. More broadly, they motivate a stage-adaptive view of reasoning efficiency, where computation adapts to the evolving functional needs of reasoning rather than simply continuing or stopping.

AI USE STATEMENT

Generative AI assistants were used to support manuscript planning, literature retrieval, mathematical exposition, English drafting, coding assistance, and preparation of tables from recorded experimental results. All AI-assisted content, code, and analyses were carefully reviewed, verified, and revised by the authors before inclusion in the manuscript. As part of the experimental pipeline, LLMs generated reasoning trajectories and responses, provided rubric-based response-quality scores, and assisted with behavioral annotation of reasoning trajectories. The authors take responsibility for the final content of this work, including all AI-assisted text, code, analyses, and experimental artifacts.

ETHICS STATEMENT

This work studies efficient reasoning on existing mathematical and scientific benchmarks. The human evaluation involved external participants who were informed about the study and participated voluntarily. Participants assessed the relative quality of model-generated mathematical responses, following the protocol described in Appendix B.2.3. We report cross-judge analyses and human validation to assess the reliability of automated quality evaluation.

REPRODUCIBILITY STATEMENT

Section 3 specifies the model architecture and training objectives, with implementation details and theoretical proofs in Appendices C and D. Measurement and validation protocols are provided in Appendix B. Section 4.1 and Appendix E describe the datasets, evaluation metrics, and experimental settings. Latent-mode analyses and latency measurements are documented in Appendices F and G. The complete rubric and classification prompts are provided in Appendices I and J.

REFERENCES

Anum Afzal, Florian Matthes, Gal Chechik, and Yftah Ziser. Knowing before saying: Llm representations encode information about chain-of-thought success before completion. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 12791–12806, 2025.

Xinghao Chen, Chak Tou Leong, Wenjin Guo, Jian Wang, Wenjie Li, and Xiaoyu Shen. What makes effective supervision in latent chain-of-thought: An information-theoretic analysis. *arXiv preprint arXiv:2606.20075*, 2026.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, et al. Do not think that much for 2+ 3=? on the overthinking of long reasoning models. In *Forty-second International Conference on Machine Learning*, 2025.

Jie Deng, Shining Liang, Jun Li, Hongzhi Li, and Yutao Xie. Conpress: Learning efficient reasoning from multi-question contextual pressure. *arXiv preprint arXiv:2602.01472*, 2026.

Yichao Fu, Junda Chen, Siqi Zhu, Zheyu Fu, Zhongdongming Dai, Yonghao Zhuang, Yian Ma, Aurick Qiao, Tajana Rosing, Ion Stoica, and Hao Zhang. Efficiently scaling LLM reasoning with Certaindex. *arXiv preprint arXiv:2412.20993*, 2025. URL https://arxiv.org/abs/2412.20993.

Weixin Guan, Liang Li, Jiapeng Liu, Bing Li, Peng Fu, Chengyang Fang, Xiaoshuai Hao, Can Ma, and Weiping Wang. Mitigating overthinking in large reasoning language models via reasoning path deviation monitoring. *arXiv preprint arXiv:2603.14251*, 2026.

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.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024.

Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, et al. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 138306–138322, 2026.

Seungone Kim, Jay Shin, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Ryan Shin, Sungdong Kim, James Thorne, Minjoon Seo, et al. Prometheus: Inducing fine-grained evaluation capability in language models. In *International Conference on Learning Representations*, volume 2024, pp. 29927–29962, 2024.

Zeping Li, Hongru Wang, Yiwen Zhao, Guanhua Chen, Yixia Li, Keyang Chen, Yixin Cao, Guangnan Ye, Hongfeng Chai, and Zhenfei Yin. Rethinking the role of entropy in optimizing tool-use behaviors for large language model agents. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27955–27967, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1288. URL https://aclanthology.org/2026.acl-long.1288/.

Jane SM Lidstone, Elizabeth Meins, and Charles Fernyhough. The roles of private speech and inner speech in planning during middle childhood: Evidence from a dual task paradigm. *Journal of Experimental Child Psychology*, 107(4):438–451, 2010.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

Xin Liu and Lu Wang. Answer convergence as a signal for early stopping in reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 17907–17918, 2025.

Dehai Min, Giovanni Vaccarino, Huiyi Chen, Yongliang Wu, Gal Yona, and Lu Cheng. Stop when reasoning converges: Semantic-preserving early exit for reasoning models. *arXiv preprint arXiv:2605.17672*, 2026.

Akira Miyake, Michael J Emerson, Francisca Padilla, and Jeung-chan Ahn. Inner speech as a retrieval aid for task goals: The effects of cue type and articulatory suppression in the random task cuing paradigm. *Acta psychologica*, 115(2-3):123–142, 2004.

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. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. Codi: Compressing chain-of-thought into continuous space via self-distillation. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 677–693, 2025.

Wenhui Tan, Fiorenzo Parascandolo, Enver Sangineto, Jianzhong Ju, Zhenbo Luo, Qian Cao, Rita Cucchiara, Ruihua Song, and Jian Luan. Restoring exploration after post-training: Latent exploration decoding for large reasoning models. *arXiv preprint arXiv:2602.01698*, 2026.

Ningning Wang, Xavier Hu, Pai Liu, He Zhu, Yue Hou, Heyuan Huang, Shengyu Zhang, Jian Yang, Jiaheng Liu, Ge Zhang, et al. Efficient agents: Building effective agents while reducing cost. *arXiv preprint arXiv:2508.02694*, 2025.

Bin Wu, Edgar Meij, and Emine Yilmaz. A joint optimization framework for enhancing efficiency of tool utilization in llm agents. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 22361–22373, 2025.

Junhong Wu, Jinliang Lu, Zixuan Ren, Gangqiang Hu, Zhi Wu, Dai Dai, et al. Llms are single-threaded reasoners: Demystifying the working mechanism of soft thinking. In *International Conference on Learning Representations*, volume 2026, pp. 4533–4548, 2026.

Shijie Xia, Xuefeng Li, Yixin Liu, Tongshuang Wu, and Pengfei Liu. Evaluating mathematical reasoning beyond accuracy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 27723–27730, 2025.

Ting Xu, Xu He, Yupu Lu, Jiankai Sun, Dong Li, Wai Lam, and Jianye Hao. Unveiling the entropy dynamics of chain-of-thought reasoning. *arXiv preprint arXiv:2606.02020*, 2026.

Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. Softcot: Soft chain-of-thought for efficient reasoning with llms. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23336–23351, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Bangji Yang, Hongbo Ma, Jiajun Fan, and Ge Liu. Batched contextual reinforcement. In *Forty-third International Conference on Machine Learning*, 2026a.

Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Minghui Chen, Zheng Lin, and Weipinng Wang. Dynamic early exit in reasoning models. In *International Conference on Learning Representations*, volume 2026, pp. 88170–88210, 2026b.

Seonghyeon Ye, Doyoung Kim, Sungdong Kim, Hyeonbin Hwang, Seungone Kim, Yongrae Jo, James Thorne, Juho Kim, and Minjoon Seo. Flask: Fine-grained language model evaluation based on alignment skill sets. In *International Conference on Learning Representations*, volume 2024, pp. 55361–55414, 2024.

Anqi Zhang, Yulin Chen, Jane Pan, Chen Zhao, Aurojit Panda, Jinyang Li, and He He. Reasoning models know when they're right: Probing hidden states for self-verification. *arXiv preprint arXiv:2504.05419*, 2025a.

Duzhen Zhang, Zhong-Zhi Li, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Xiuyi Chen, Yingying Zhang, et al. From system 1 to system 2: A survey of reasoning large language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025b.

Zhen Zhang, Xuehai He, Weixiang Yan, Ao Shen, Chenyang Zhao, and Xin Wang. Soft thinking: Unlocking the reasoning potential of llms in continuous concept space. *Advances in Neural Information Processing Systems*, 38:168990–169012, 2026.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 10495–10516, 2025c.

Haoran Zhao, Yuchen Yan, Yongliang Shen, Haolei Xu, Wenqi Zhang, Kaitao Song, Jian Shao, Weiming Lu, Jun Xiao, and Yueting Zhuang. Let lrms break free from overthinking via self-braking tuning. *Advances in Neural Information Processing Systems*, 38:1861–1887, 2026a.

Yiwen Zhao, Zhihao Wen, Yuchen Mao, Mingxuan Jiang, Yihao Hu, Pan Wang, Xin Zhang, and Wei Wu. Towards better agents for multi-turn user interaction: The next user turn is more than context. *arXiv preprint arXiv:2608.17499*, 2026b.

A Related Work

A.1 Efficient Reasoning and Early Exit

Computational efficiency has received growing attention in LLM agents (Li et al., 2026; Zhao et al., 2026b; Wang et al., 2025; Wu et al., 2025). This concern is particularly acute for LRMs: their strong performance relies on extended test-time reasoning, yet long chains of thought frequently contain computation whose marginal utility diminishes as generation proceeds.

To reduce this overhead, inference-time methods dynamically terminate generation when further computation is deemed unnecessary. Dynasor allocates computation using certainty-based measures of reasoning progress (Fu et al., 2025), Dynamic Early Exit relies on trial-answer confidence to determine stopping points (Yang et al., 2026b), PUMA detects semantic redundancy between successive steps (Min et al., 2026), reasoning-path deviation monitors high-entropy transitions to identify overthinking (Guan et al., 2026), and entropy-based change-point detection separates exploratory and converged reasoning phases (Xu et al., 2026). These methods ultimately decide whether explicit reasoning should *continue* or *terminate*. In contrast, we distinguish answerability from response readiness and identify a non-trivial intermediate regime where the solution is recoverable but additional computation still improves final quality. This motivates a third option beyond stopping or continuing: preserving the essential post-answerability computation through compact latent consolidation while bypassing expensive natural-language realization.

A.2 Latent and Continuous Reasoning

Latent-reasoning methods explore whether intermediate computation must be verbalized in natural language. Coconut recursively feeds hidden states as continuous thoughts (Hao et al., 2024), CODI distills explicit CoT into continuous representations (Shen et al., 2025), and SoftCoT generates instance-specific soft thoughts through a lightweight auxiliary model (Xu et al., 2025). However, fully implicit reasoning can suffer from degraded reasoning fidelity or representational drift (Shen et al., 2025; Chen et al., 2026). This motivates selective latent reasoning, which converts locally predictable spans into latent representations while keeping uncertain or critical steps explicit. We ground the handoff to latent computation in the functional transition from solution discovery to response consolidation. BRIDGE is supervised using the Solution–Response Gap, while SCOUT estimates when to invoke it online. The handoff threshold controls the transition between explicit reasoning and latent consolidation, as examined in RQ3.

B Solution–Response Gap: Measurement Details and Robustness

B.1 Measurement Details

B.1.1 Trajectory Probing

As described in Section 2.1, we segment each natural reasoning trajectory at "`\n\n`" boundaries. At each boundary, we retain the original prompt and truncated reasoning prefix, append only `</think>`, and sample $K = 4$ final responses at non-zero temperature without any additional answer-extraction instruction. The final answer extracted from each response is normalized for whitespace and equivalent LaTeX forms and compared with the gold answer. The resulting correctness labels determine C_i. We use the reliability threshold $\tau_A = 3/4$, such that a boundary is considered answerable when at least three of the four responses are correct.

The same four responses are reused for response-quality evaluation, with Q_i computed as the mean rubric score over the correct responses. No additional responses are generated for quality evaluation, and correctness labels are not supplied to the rubric judge.

B.1.2 Rubric Scoring

The rubric instantiated in Section 2.1 contains four defect-sensitive dimensions, summarized in Table 3. Each dimension is scored on an integer scale from 0 to 4, with higher scores indicating fewer defects. Scoring is performed deterministically with temperature 0 and JSON-constrained output,

and malformed outputs are retried until all four dimensions are valid integers in $[0, 4]$. The complete system prompt provided to the judge models appears in Appendix I.

Table 3: Rubric dimensions for final-response quality. Each dimension is scored from 0 to 4, with higher scores indicating fewer defects.

Dimension	Evaluation criterion
Answer convergence	Whether the response presents a settled solution without visible reconsideration, answer switching, repeated re-derivation, or post-conclusion repair.
Reasoning directness	Whether the response proceeds efficiently toward its conclusion without abandoned branches, unnecessary alternative derivations, or repeated setup.
Non-progressive redundancy	Whether the response avoids repeating, re-verifying, or rephrasing already-established content without adding reasoning value.
Local coherence	Whether the response remains internally consistent and avoids abrupt logical transitions, unexplained jumps, or contradictions.

B.1.3 Quality Aggregation and Boundary Selection

At each boundary b_i, we compute C_i and Q_i from the same $K = 4$ final responses sampled by retaining the original prompt and truncated reasoning prefix and appending only `</think>`. For response y_{ij}, let score_{ij} denote the mean of its four rubric scores. The response quality is then defined as

$$Q_i = \frac{1}{|\mathcal{S}_i|} \sum_{j \in \mathcal{S}_i} \text{score}_{ij}, \qquad \mathcal{S}_i = \{j \in \{1, \ldots, K\} : y_{ij} \text{ is correct}\}. \tag{12}$$

Thus, C_i measures correctness over all K responses, whereas Q_i measures the mean quality of their correct subset. Three edge cases complete the operational definition. First, if all K sampled responses at a boundary are incorrect, we set $Q_i = 0$. Second, if a correct response lacks a valid rubric score, that boundary is treated as having incomplete quality coverage and is excluded from readiness selection. Third, if response quality never strictly improves after t_A, no Solution–Response Gap is recorded for that trajectory. When several later boundaries attain the same maximum quality, we select the earliest one as t_R. Let ℓ_i denote the token offset associated with boundary b_i. The indices t_A and t_R identify reasoning boundaries, and for all empirical results reported, we measure the gap as

$$\Delta_{\text{SR}}^{\text{token}} = \ell_{t_R} - \ell_{t_A}. \tag{13}$$

B.2 Cross-Judge Robustness

Because t_R is determined from rubric scores, its exact location may depend on the judge. We therefore evaluate the Solution–Response Gap under GPT-4o-mini, DeepSeek-V4-Flash, and Qwen3.5-Flash while holding the reasoning trajectories, sampled final responses, and correctness labels fixed. This three-judge analysis is used to assess cross-judge robustness. Based on the human validation in Section B.2.3, we use Qwen3.5-Flash for training-time supervision and DeepSeek-V4-Flash as the independent rubric judge for downstream evaluation.

B.2.1 Eligibility Across Judges

Table 4 reports the evaluation cohorts for Qwen3-4B-Thinking. The answerability criterion itself is judge-independent, whereas the number of trajectories with an observed gap varies because different judges need not identify a strict post-answerability quality improvement on exactly the same problems.

Each judge detects post-answerability quality improvement across all three benchmarks, although the prevalence varies across judges. For Figures 2, 8, and 9, each judge is evaluated on its own eligible set, consisting of trajectories for which that judge identifies a valid post-answerability response-readiness point. This judge-specific evaluation preserves the complete set of valid observations available under each scoring criterion and allows us to assess whether the Solution–Response Gap consistently emerges across heterogeneous rubric judges.

Table 4: Evaluation cohorts for Qwen3-4B-Thinking. "Probe coverage" denotes trajectories with answerability measurements; "usable t_A" additionally requires a defined answerability point with valid response-quality coverage; "gap observed" denotes trajectories exhibiting a strict post-answerability improvement in response quality.

Dataset	Judge	Probe coverage	Usable t_A	Gap observed
MATH-500	GPT-4o-mini	469	462	355
	DeepSeek-V4-Flash	469	465	239
	Qwen3.5-Flash	469	462	199
AIME 2024	GPT-4o-mini	25	24	19
	DeepSeek-V4-Flash	25	25	12
	Qwen3.5-Flash	25	24	12
AIME 2025	GPT-4o-mini	24	23	17
	DeepSeek-V4-Flash	24	23	17
	Qwen3.5-Flash	24	23	16

B.2.2 Judge Agreement and Calibration

Across approximately 17,300 commonly scored MATH-500 final responses, Pearson correlations are 0.54 for GPT-4o-mini versus DeepSeek-V4-Flash, 0.49 for GPT-4o-mini versus Qwen3.5-Flash, and 0.88 for DeepSeek-V4-Flash versus Qwen3.5-Flash. The corresponding Spearman correlations are 0.53, 0.45, and 0.75. The judges also differ in absolute calibration, with mean rubric scores of 2.81, 3.72, and 3.75, respectively. These results indicate substantially stronger agreement between DeepSeek-V4-Flash and Qwen3.5-Flash than between either judge and GPT-4o-mini. We next examine whether these differences are also reflected in human assessments.

B.2.3 Human Validation and Judge Selection

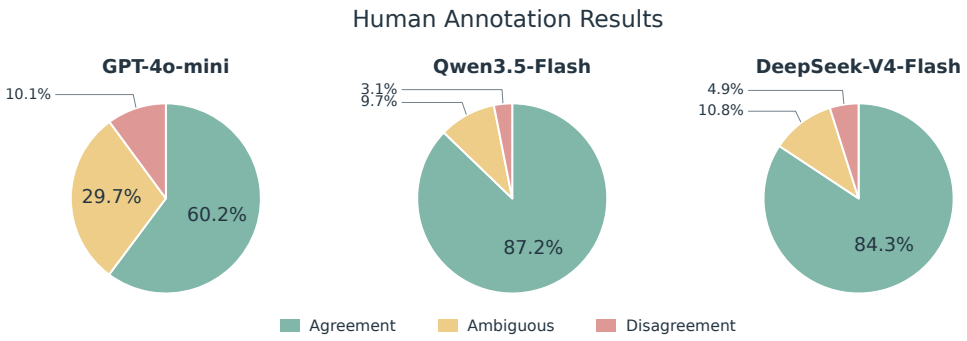

Figure 7: Human validation of rubric judges. Human annotators evaluate the quality ordering of paired final responses generated at the answerability and response-readiness boundaries, t_A and t_R. For each rubric judge, we report the proportion of cases in which its ordering agrees with the human assessment, is considered ambiguous due to an insufficiently clear quality difference, or disagrees with the human assessment. Qwen3.5-Flash and DeepSeek-V4-Flash exhibit substantially higher agreement and lower ambiguity than GPT-4o-mini.

To further assess judge reliability, we conduct a human evaluation on paired final responses generated at the corresponding answerability and response-readiness boundaries, t_A and t_R, on MATH-500. For each rubric judge, we randomly sample 150 response pairs from its positive-gap cohort. Each pair is independently evaluated by 10 human annotators, yielding 1,500 human judgments per judge. Annotators assess the quality ordering produced by each rubric judge as *agreement*, when it is consistent with the human assessment; *ambiguous*, when the response pair does not exhibit a sufficiently clear quality difference to support the judge's ordering; or *disagreement*, when the judge's ordering conflicts with the human assessment. For each annotator, we compute the proportions of agreement, ambiguity, and disagreement over the 150 response pairs, and report the macro-average across annotators.

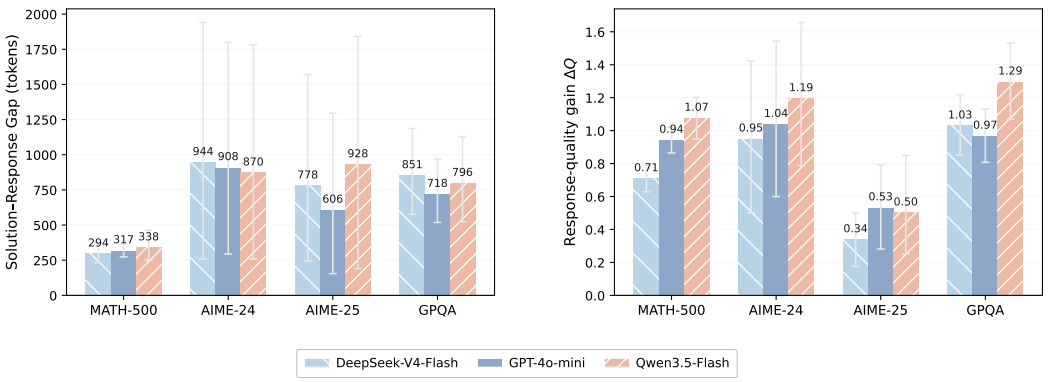

Figure 8: Solution–Response Gap for the 1.5B reasoning model. We repeat the measurement in Figure 2 with the smaller reasoning model under the same three rubric judges. The figure reports the mean token distance between answerability and response readiness together with the corresponding gain in final-response quality. Bars denote means and error bars indicate 95% confidence intervals.

As shown in Figure 7, GPT-4o-mini exhibits substantially weaker alignment with human judgments, with 60.2% agreement, 29.7% ambiguous cases, and 10.1% disagreement. In contrast, Qwen3.5-Flash achieves 87.2% agreement, with 9.7% ambiguity and 3.1% disagreement, while DeepSeek-V4-Flash achieves 84.3% agreement, with 10.8% ambiguity and 4.9% disagreement. The human evaluation is therefore consistent with the judge-to-judge analysis above: Qwen3.5-Flash and DeepSeek-V4-Flash provide substantially more reliable fine-grained response-quality assessments than GPT-4o-mini.

Despite these calibration differences, all three judges identify post-answerability response-quality improvements, providing a cross-judge robustness check for the existence of the Solution–Response Gap. Based on the human validation, we retain Qwen3.5-Flash and DeepSeek-V4-Flash for distinct roles in the remaining pipeline: Qwen3.5-Flash is used to construct training-time response-quality supervision, whereas DeepSeek-V4-Flash is used as the independent rubric judge for downstream method evaluation. This separation avoids using the same rubric judge for both constructing the training supervision and evaluating the resulting method. GPT-4o-mini is retained only in robustness analyses of the Solution–Response Gap.

B.3 Cross-Model Robustness

To establish that the Solution–Response Gap is a general phenomenon across LRMs, we repeat the measurement pipeline on additional models while keeping trajectory segmentation, `</think>`-only final-response sampling, correctness evaluation, quality aggregation, and boundary selection unchanged.

Figures 8 and 9 extend the analysis beyond the Qwen3-4B-Thinking model used in the main text. The Solution–Response Gap is observed across the evaluated models and benchmarks. Its duration and associated response-quality gain vary across models and tasks.

B.4 Robustness to Alternative Operationalizations

To examine whether the Solution–Response Gap depends on a particular operational definition of t_A or t_R, we recompute the gap under four alternative settings while keeping the underlying trajectories, sampled final responses, correctness labels, and rubric scores fixed. Our default setting is described in Section 2.1.

We consider the following variants:

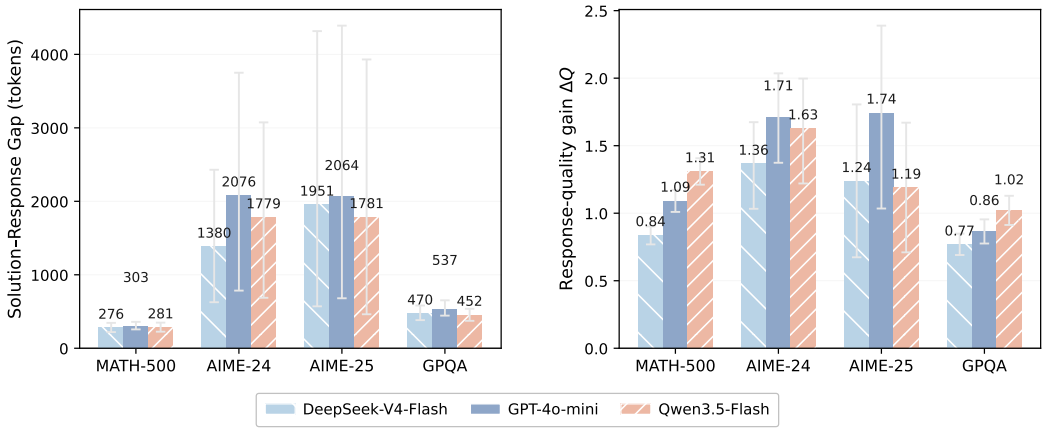

Figure 9: Solution–Response Gap for the 7B reasoning model. The same evaluation is repeated with the larger reasoning model, using the measurement protocol and rubric judges from Figure 2. Bars denote means and error bars indicate 95% confidence intervals.

- **Consecutive t_A.** Instead of declaring the first boundary with $C_i \geq 3/4$ answerable, we require two consecutive boundaries to satisfy the same threshold. Specifically,

$$t_A = \min\left\{i : C_i \geq \frac{3}{4} \text{ and } C_{i+1} \geq \frac{3}{4}\right\}.$$

  This variant imposes an additional stability requirement on answerability and therefore tends to move t_A later.

- $2/4$ **answerability threshold.** We retain the single-boundary definition but lower the reliability threshold from $3/4$ to $2/4$. This more permissive criterion allows answerability to be detected from weaker evidence and therefore tends to move t_A earlier.

- **Zero-fill incorrect responses.** The default definition computes Q_i only over the correct responses within the shared set of $K = 4$ samples. In this variant, all four responses from the same set are included in the average, with every incorrect response assigned a quality score of zero:

$$Q_i^{\text{zero}} = \frac{1}{K}\sum_{j=1}^{K} \mathbf{1}[y_{ij} \text{ is correct}]\ \text{score}_{ij}.$$

  This incorporates answer reliability into the quality score, so a boundary is penalized when it produces high-quality correct responses only inconsistently.

- **Consecutive $0.9Q^\star$ readiness.** Instead of requiring t_R to attain the strict maximum $Q^\star$, we use a more permissive readiness threshold while requiring local stability. Specifically, t_R is the earliest post-answerability boundary for which both the current and the next boundary reach at least 90% of the best observed quality:

$$t_R = \min\left\{i \geq t_A : Q_i \geq 0.9Q^\star \text{ and } Q_{i+1} \geq 0.9Q^\star\right\}.$$

  Because near-optimal responses are accepted rather than requiring the exact maximum, this criterion generally identifies response readiness earlier than the default definition.

Figure 10 compares these variants with the default operationalization.

Each operationalization produces a positive Solution–Response Gap, with its magnitude reflecting the chosen boundary and quality criteria. Requiring consecutive successful boundaries delays t_A and generally shortens the gap, whereas lowering the answerability threshold to $2/4$ moves t_A earlier and tends to increase it. Zero-filling incorrect responses similarly changes the estimated response-quality trajectory by incorporating answer reliability into Q_i.

With consecutive $0.9Q^\star$ readiness, accepting near-optimal responses tends to shift t_R earlier and shorten Δ_{SR}. The resulting gaps span several hundred reasoning tokens across all three benchmarks.

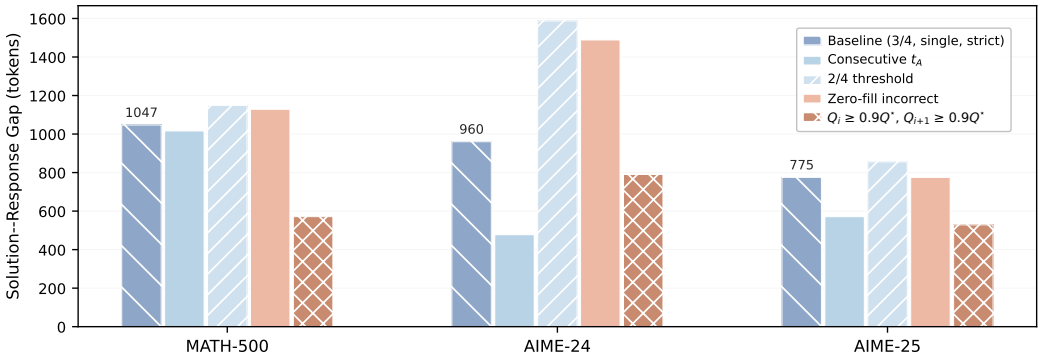

Figure 10: Robustness of the Solution–Response Gap to alternative operationalizations. We vary the definition of answerability, the aggregation of response quality, and the criterion for response readiness while keeping the underlying trajectories and sampled responses fixed. The estimated gap length varies across definitions, with positive gaps observed on MATH-500, AIME 2024, and AIME 2025.

Across these operationalizations, answerability precedes response readiness, while the estimated gap length varies with the measurement criterion. The default definition measures answer reliability and conditional response quality from the same sampled responses, identifying response readiness as the earliest post-answerability boundary that attains the highest observed response quality.

B.5 Validation with Independently Generated Responses

The default measurement uses sampled responses both to identify the response-ready boundary and to quantify its quality advantage. Selecting the highest observed quality among subsequent boundaries can produce optimistic estimates under finite response sampling. We therefore examine whether the quality ordering at the selected boundaries persists on a second, independently generated response batch.

Evaluation protocol. We evaluate Qwen3-4B-Thinking using Qwen3.5-Flash and DeepSeek as rubric judges. Each judge defines its own positive-gap cohort from the original evaluation, comprising 199 trajectories for Qwen3.5-Flash and 239 for DeepSeek. The original quality gain is strictly positive for every trajectory in its respective cohort by construction. For each judge and trajectory, we freeze the original answerability boundary t_A and response-readiness boundary t_R before generating the second response batch.

At each frozen boundary, we retain the original prompt and reasoning prefix, append the end-of-thinking marker, and generate fresh final responses using independent decoding randomness. We apply the same rubric and correct-response quality aggregation as in Appendix B.1.3. Neither boundary is reselected using the new responses. Letting $Q_i^{(2)}$ denote the quality estimate from the second batch at boundary i, we compute

$$\Delta Q^{(2)} = Q_{t_R}^{(2)} - Q_{t_A}^{(2)}. \tag{14}$$

We classify each trajectory as retaining a positive gain ($\Delta Q^{(2)} > 0$), exhibiting a tie ($\Delta Q^{(2)} = 0$), or showing a reversal ($\Delta Q^{(2)} < 0$). We additionally report the non-decreasing proportion ($\Delta Q^{(2)} \geq 0$), which includes both positive gains and ties. All trajectories in each original cohort remain in the evaluation, including those with tied or reversed quality ordering.

Results. As shown in Table 5, strictly positive quality gains persist for 173 of 199 trajectories (86.9%) under Qwen3.5-Flash and 201 of 239 trajectories (84.1%) under DeepSeek. Including ties, the corresponding non-decreasing proportions are 95.5% and 94.1%, respectively. Reversals occur in 4.5% of the Qwen3.5-Flash cohort and 5.9% of the DeepSeek cohort. Thus, under both judges, the response-quality advantage at the selected later boundary persists on fresh responses for the majority of initially detected positive-gap trajectories.

Table 5: Independent validation of the response-quality ordering at fixed boundaries under two rubric judges. Each row reports results on that judge's original positive-gap cohort of size N. Non-decreasing includes both positive gains and ties. Percentages are rounded to one decimal place.

Judge	N	Positive	Tied	Reversed	Non-decreasing
Qwen3.5-Flash	199	173 (86.9%)	17 (8.5%)	9 (4.5%)	190 (95.5%)
DeepSeek	239	201 (84.1%)	24 (10.0%)	14 (5.9%)	225 (94.1%)

These results support the reproducibility of post-answerability quality improvements at the previously identified boundaries. The reported replication rates are conditional on each judge's original positive-gap cohort and quantify the persistence of those observations under independent response generation. Together, the high replication rates across both judges support the stability of the quality advantage associated with the detected Solution–Response Gap.

C SCOUT TRAINING AND THRESHOLD SELECTION DETAILS

C.1 ONLINE STATE AGGREGATION

SCOUT makes online handoff decisions from the hidden states produced by the ongoing reasoning trajectory. At each completed reasoning step, we pool token states separately at the four selected layers and form the state and transition features in Equation 5. The first step uses a zero transition feature. The implementation detects paragraph separators at the token level and accumulates steps shorter than eight tokens into the following segment before making a new decision. This avoids treating a very short fragment as a complete reasoning step. Once a segment is scored, its pooled state becomes the preceding-step state for the next decision. Feature extraction maintains the current segment's running sums and the preceding pooled state, updating them after each boundary.

C.2 INTERVAL SUPERVISION AND CENSORING

SCOUT predicts a conditional stopping probability $\widehat{h}_i$ at each completed step, as defined in Equation 6. For an annotated trajectory $\mathcal{R}$, we order the probed boundaries $\mathcal{B}_{\mathcal{R}}$ and identify the target interval $\mathcal{I}_{\mathcal{R}} = \{i \in \mathcal{B}_{\mathcal{R}} : t_A \le i \le t_R\}$. The answerable boundary uses the $3/4$ probe-reliability criterion, and the response-ready boundary uses the highest observed quality among subsequent correct responses. Equation 7 assigns each boundary the probability that it is the first stop and maximizes the total probability of first stopping within this interval. The survival factor $\prod_{j<i}(1-\widehat{h}_j)$ couples the decisions across steps: stopping before t_A reduces the probability available to the target interval, while continuing beyond t_R leaves that interval's probability mass unrealized.

When interval annotations are incomplete, we use the observed prefix known to precede answerability to supervise continuation. Let $\mathcal{U}_{\mathcal{R}}$ denote these observed pre-answerability boundaries. Their continuation loss is

$$\ell_{\mathcal{R}}^{\text{cont}} = -\log\left(\prod_{i\in\mathcal{U}_{\mathcal{R}}}(1-\widehat{h}_i)\right) = -\sum_{i\in\mathcal{U}_{\mathcal{R}}}\log(1-\widehat{h}_i). \tag{15}$$

Unobserved boundaries do not receive a stopping target. We average the applicable interval or continuation loss equally across training trajectories, keeping the reasoner and BRIDGE frozen. Data are partitioned at the problem level, with all boundaries from a problem assigned to the same split. After training, the SCOUT parameters remain fixed during operating-point selection.

C.3 SELECTING AN OPERATING POINT

For a candidate threshold S, we replay the first crossing of the cumulative first-stop probability $\widehat{F}_i = 1 - \prod_{j\le i}(1-\widehat{h}_j)$ on each trajectory in the validation set $\mathcal{D}_{\text{val}}$. If no boundary crosses S, the policy retains the natural reasoning trajectory. We measure its mean retained explicit-token count, denoted $\widehat{T}_{\text{exp}}(S)$, and its empirical handoff-error frequency. Consistent with the answerability

definition, we count a handoff as an error when fewer than three quarters of its answer probes are correct:

$$\widehat{E}(S) = \frac{1}{|\mathcal{D}_{\text{val}}|} \sum_{\mathcal{R} \in \mathcal{D}_{\text{val}}} \mathbf{1}\left[\widehat{t}_S(\mathcal{R}) \text{ exists}\right] \mathbf{1}\left[C_{\widehat{t}_S(\mathcal{R})} < \tfrac{3}{4}\right], \qquad S^\star \in \arg\min_{S:\widehat{E}(S)\le\epsilon} \widehat{T}_{\exp}(S). \quad (16)$$

The denominator includes all validation trajectories, including those without a handoff. With $\epsilon = 0.05$, this procedure selects $S^\star = 0.9147$, reported as 0.91, on $\mathcal{D}_{\text{val}}$. At this operating point, SCOUT triggers on 259 of the 400 validation trajectories and produces 19 handoff errors, corresponding to $\widehat{E}(S^\star) = 4.75\%$. Appendix E.5 reports supplementary threshold operating points.

C.4 THEORETICAL ANALYSIS OF SCOUT

Main conclusion. On a fixed sequence of decision boundaries, a sufficiently small interval loss guarantees that the first threshold crossing occurs between the annotated t_A and t_R. Across a collection of trajectories, the mean interval loss also provides an upper bound on the fraction that miss this window. These results connect the quantity minimized during training to the desired event: a first handoff within the annotated answerability–readiness interval.

Setup and interval objective. Fix a trajectory with a nonempty annotated target window and an ordered sequence of decision boundaries, relabeled as $1, \ldots, L$. Let $[a, b]$ index the selected boundaries between the measured t_A and t_R. The loss and threshold crossing in the following statements are computed on this same sequence. For the probed sequence $\mathcal{B}_{\mathcal{R}}$, this gives a result about probed-boundary replay; applying it to the full online sequence requires evaluating the interval loss on that full sequence. All logarithms are natural. For predicted hazards $\widehat{h}_j \in (0, 1)$, define the model-assigned survival and cumulative probabilities by

$$V_0 = 1, \qquad V_j = \prod_{k=1}^{j} (1 - \widehat{h}_k), \qquad q_j = V_{j-1}\widehat{h}_j, \qquad \widehat{F}_j = 1 - V_j. \quad (17)$$

Here, V_j is the probability assigned to surviving through boundary j, q_j is the probability assigned to a first stop at j, and $\widehat{F}_j$ is the cumulative stopping probability. This is the discrete-time hazard representation of a first stopping time.[1] Since $q_j = V_{j-1} - V_j$, the probability assigned to the target window telescopes:

$$m_{\mathcal{R}} = \sum_{j=a}^{b} q_j = V_{a-1} - V_b = \widehat{F}_b - \widehat{F}_{a-1} = \exp(-\ell_{\mathcal{R}}). \quad (18)$$

Equivalently, the interval loss decomposes as

$$\ell_{\mathcal{R}} = -\sum_{j<a} \log(1 - \widehat{h}_j) - \log\left(1 - \prod_{j=a}^{b} (1 - \widehat{h}_j)\right). \quad (19)$$

The two terms penalize stopping mass before the annotated window and insufficient conditional stopping mass inside it, respectively, while leaving the exact in-window location unspecified. Thus, minimizing the interval loss concentrates first-stopping probability in the target window. The following proposition shows when this concentration also guarantees an in-window handoff under the threshold rule.

Proposition C.1 (Interval localization under cumulative thresholding) *For $S \in (0, 1)$, define $\widehat{t}_S = \min\{j : \widehat{F}_j \ge S\}$, with $\widehat{t}_S = +\infty$ when no boundary crosses the threshold. Then*

$$\widehat{t}_S \in [a, b] \quad \Longleftrightarrow \quad \widehat{F}_{a-1} < S \le \widehat{F}_b. \quad (20)$$

In particular, a sufficient condition for interval localization is

$$\ell_{\mathcal{R}} < c(S), \qquad c(S) = -\log \max\{S, 1 - S\} \quad \Longrightarrow \quad \widehat{t}_S \in [a, b]. \quad (21)$$

[1] For background, see H. Kvamme and Ø. Borgan, "Continuous and discrete-time survival prediction with neural networks," *Lifetime Data Analysis*, 27:710–736, 2021. https://doi.org/10.1007/s10985-021-09532-6.

Proof. Monotonicity of $\widehat{F}_j$ gives Equation 20. By Equation 18, $\widehat{F}_{a-1} \le 1 - e^{-\ell_{\mathcal{R}}}$ and $\widehat{F}_b \ge e^{-\ell_{\mathcal{R}}}$. When $\ell_{\mathcal{R}} < c(S)$, we have $e^{-\ell_{\mathcal{R}}} > \max\{S, 1-S\}$, implying $\widehat{F}_{a-1} < S < \widehat{F}_b$ and therefore an in-window handoff. ∎

Interpretation: a guarantee for a single trajectory. Proposition C.1 gives a concrete meaning to a small interval loss. Below the threshold-dependent level $c(S)$, the cumulative stopping probability stays below S before the target window and reaches S by its end. The first handoff must therefore occur inside that window. This is a guarantee about the deterministic threshold crossing, not just the probability mass assigned to the interval. The next result extends this connection to the fraction of trajectories whose handoffs miss the window.

Corollary C.2 (Interval-miss rate) *For any collection of N trajectories with complete, nonempty target windows $[a_n, b_n]$, let $\overline{\ell}_{\text{int}} = N^{-1} \sum_{n=1}^{N} \ell_{\mathcal{R}_n}$. For the same fixed predictor and threshold on this collection, the fraction of trajectories without an in-window threshold crossing satisfies*

$$\widehat{R}_{\text{out}}(S) = \frac{1}{N} \sum_{n=1}^{N} \mathbf{1}\left[\widehat{t}_S^{(n)} \notin [a_n, b_n]\right] \le \min\left\{1, \frac{\overline{\ell}_{\text{int}}}{c(S)}\right\}, \qquad S \in (0,1). \tag{22}$$

A trajectory without a threshold crossing is counted as an interval miss.

Proof. By Equation 20, $m_{\mathcal{R}} \le 1 - S$ for early handoffs and $m_{\mathcal{R}} < S$ for late or absent handoffs. Taking negative logarithms and using the disjointness of these events gives

$$\ell_{\mathcal{R}} \ge [-\log(1-S)]\, \mathbf{1}[\widehat{t}_S < a] + [-\log S]\, \mathbf{1}[\widehat{t}_S > b] \ge c(S)\, \mathbf{1}[\widehat{t}_S \notin [a,b]]. \tag{23}$$

Averaging over the collection and using $\widehat{R}_{\text{out}}(S) \le 1$ proves the result. ∎

Interpretation: a bound for a collection of trajectories. Every early, late, or absent handoff incurs an interval loss of at least $c(S)$. Consequently, the average loss limits how many trajectories can miss their annotated windows: a smaller mean loss gives a no-larger upper bound on the miss rate at a fixed threshold. The bound becomes informative, below one, when $\overline{\ell}_{\text{int}} < c(S)$. Both the loss and the miss rate are evaluated on the same collection of trajectories. This interval-miss event is distinct from the probe-based handoff error used for calibration in Equation 16.

Implications for SCOUT. Together, these results justify the interval loss as a surrogate objective for placing the first handoff between annotated answerability and response readiness, while allowing different trajectories to hand off at different points within that window. The threshold then controls the ordering of handoffs: for fixed predicted hazards and $0 < S_1 \le S_2 < 1$, monotonicity gives $\widehat{t}_{S_1} \le \widehat{t}_{S_2}$, so a higher threshold cannot trigger an earlier handoff on the same trajectory. The guarantees concern annotated-window localization on the evaluated trajectories and boundary sequence; generalization to unseen trajectories, the best position within the window, and final-answer accuracy and response quality require separate evaluation, which we validate empirically in Section 4.

D BRIDGE ARCHITECTURE, TRAINING, AND INFERENCE

D.1 RECURRENT PROJECTOR AND GRADIENT FLOW

For a reasoner with hidden dimension d, BRIDGE uses two d-to-d linear layers, a GELU activation, and output layer normalization. The same projector is reused at every latent step. Its positive output scale is parameterized as $\rho = \exp(\eta)$ and learned jointly with the projector. The projector emits a continuous embedding in the reasoner's input space. Each embedding is consumed by the frozen reasoner; the resulting hidden state determines the next embedding.

Training pairs share the original problem but differ in the amount of explicit reasoning retained. The student starts from a boundary in $[t_A, t_R)$, while the teacher retains the response-ready prefix. Both append the same end-of-thinking marker and evaluate the same correct target response. The teacher branch is evaluated without gradients. Although the reasoner's parameters remain frozen, the student computation remains differentiable so that the response loss and state-alignment loss can

update the projector through all latent steps. The state-alignment target is taken immediately after the closing marker, before the first target-response token, avoiding dependence on the teacher-forced answer content.

D.2 Training Configuration

Table 6 records the recurrent BRIDGE configuration for Qwen3-4B-Thinking-2507. The projector has 13,117,441 trainable parameters, while the backbone is frozen. The teacher-confidence term in Equation 10 weights each target token's cross-entropy by its probability under the teacher. Together with response cross-entropy and state alignment, this term guides the latent trajectory toward the response-ready teacher state and its target response.

Table 6: Recurrent BRIDGE training settings for Qwen3-4B-Thinking-2507.

Setting	Value	Setting	Value
Hidden dimension d	2560	Latent steps K_{B}	32
Initial scale ρ	0.02	Confidence weight β	0.2
Alignment weight λ	1.0	Learning rate	10^{-4}
Training epochs	3	Warmup ratio	0.05
Gradient clipping	1.0	Effective batch size	8
Maximum sequence length	26,624	Maximum target length	4,096

D.3 Inference Procedure

Algorithm 1 summarizes inference. The measured boundaries t_A and t_R define training pairs, while SCOUT's cumulative first-stop probability determines the handoff during inference. In particular, SCOUT selects the handoff from the current state, and BRIDGE performs a fixed number of steps before final-response generation. Inference follows a single sequence of explicit reasoning, latent consolidation, and final-response generation.

Algorithm 1 Stage-adaptive inference with *SABER*

Require: Problem x, frozen reasoner π_θ, SCOUT, BRIDGE, threshold S, latent budget K_{B}

Initialize the reasoning prefix, SCOUT's segment state, $i = 0$, and $\widehat{F}_0 = 0$.
while explicit reasoning has not ended and its token budget remains **do**
  Generate the next reasoning token and update the segment state.
  if the natural end-of-thinking marker is generated **then**
    Generate the final response with π_θ and **return** .
  end if
  if a complete segment is available **then**
    Set $i \leftarrow i + 1$ and compute $\widehat{h}_i$ using Equations 5–6.
    Update $\widehat{F}_i \leftarrow \widehat{F}_{i-1} + (1 - \widehat{F}_{i-1})\widehat{h}_i$.
    if $\widehat{F}_i \geq S$ **then**
      for $k = 1, \ldots, K_{\mathrm{B}}$ **do**
        Produce and consume $z_{i,k}$ using Equation 8.
      end for
      Append c, generate the final response with π_θ, and **return** .
    end if
  end if
end while
If the reasoning budget is exhausted, append c and generate the final response.

D.4 Theoretical Analysis of Bridge

Main conclusion. Under a shared linear output head and a common positive decoding temperature, the squared distance between the two pre-response states bounds the KL divergence between

their first-response-token distributions, up to a factor determined by the readout and temperature. This bound separates state-norm mismatch from directional mismatch. At fixed state norms, reducing the cosine alignment loss decreases the directional contribution to the bound; under the additional equal-norm condition, the bound is proportional to the cosine loss alone. The result gives an output-level interpretation of the state-alignment objective.

Setup and shared readout. We analyze the alignment term in Equation 10 at the transition from latent consolidation to final-response generation. The following result assumes that the aligned states $u_B, u_R \in \mathbb{R}^d$ are the nonzero inputs to the reasoner's shared linear output head after processing the end-of-thinking marker c, including any final normalization preceding that head. Here, u_B is the state reached after BRIDGE's latent consolidation, and u_R is the corresponding response-ready teacher state. Let W_{out} and b_{out} be that head's weight matrix and bias. All logarithms are natural. At a common temperature $\tau_{\text{dec}} > 0$, define the full-vocabulary distributions before decoding filters as

$$\xi_X = \frac{W_{\text{out}} u_X + b_{\text{out}}}{\tau_{\text{dec}}}, \qquad p_X^{(1)} = \text{softmax}(\xi_X), \qquad X \in \{B, R\}. \tag{24}$$

These distributions predict the first final-response token from the student and response-ready teacher states, respectively. Write $r_X = \|u_X\|_2$ and $d_{\cos} = 1 - u_B^\top u_R/(r_B r_R)$.

Proposition D.1 (First-token KL bound under a shared readout) *Under the shared readout in Equation 24,*

$$D_{\text{KL}}\left(p_R^{(1)} \,\|\, p_B^{(1)}\right) \le \frac{\|W_{\text{out}}\|_2^2}{4\tau_{\text{dec}}^2}\left[(r_B - r_R)^2 + 2 r_B r_R\, d_{\cos}\right], \tag{25}$$

where $\|W_{\text{out}}\|_2$ denotes the spectral norm. In the equal-norm case $r_B = r_R = r$, this specializes to

$$D_{\text{KL}}\left(p_R^{(1)} \,\|\, p_B^{(1)}\right) \le \frac{r^2 \|W_{\text{out}}\|_2^2}{2\tau_{\text{dec}}^2}\left(1 - \frac{u_B^\top u_R}{\|u_B\|_2 \|u_R\|_2}\right). \tag{26}$$

Proof. Let $A(\xi) = \log \sum_v \exp(\xi_v)$, so that $\nabla A(\xi) = p = \text{softmax}(\xi)$ and $\nabla^2 A(\xi) = \text{diag}(p) - pp^\top$. This symmetric matrix has absolute row sums $2p_v(1 - p_v) \le 1/2$, so its spectral norm is at most $1/2$. This gives the standard $1/2$ smoothness bound for log-sum-exp.[2] With $\Delta\xi = \xi_B - \xi_R$, the softmax KL divergence is the associated Bregman divergence:

$$\begin{aligned} D_{\text{KL}}\left(p_R^{(1)} \,\|\, p_B^{(1)}\right) &= A(\xi_B) - A(\xi_R) - \nabla A(\xi_R)^\top \Delta\xi \\ &= \int_0^1 (1-t)\, \Delta\xi^\top \nabla^2 A(\xi_R + t\Delta\xi)\, \Delta\xi \, dt \\ &\le \frac{1}{4}\|\Delta\xi\|_2^2 \le \frac{\|W_{\text{out}}\|_2^2}{4\tau_{\text{dec}}^2}\|u_B - u_R\|_2^2. \end{aligned} \tag{27}$$

Finally,

$$\|u_B - u_R\|_2^2 = r_B^2 + r_R^2 - 2u_B^\top u_R = (r_B - r_R)^2 + 2 r_B r_R\, d_{\cos}. \tag{28}$$

Substituting this identity proves Equation 25; setting $r_B = r_R$ gives Equation 26. ∎

Interpretation: what state alignment controls. Proposition D.1 connects alignment in representation space to a discrepancy in the model's output distribution at the start of the response. The two terms in Equation 25 have distinct meanings: $(r_B - r_R)^2$ measures a difference in state magnitudes, whereas $2r_B r_R d_{\cos}$ measures a difference in state directions. For fixed state norms, a smaller cosine loss therefore gives a tighter upper bound on the first-token KL divergence. When the norms are equal, the magnitude term vanishes, so cosine distance alone controls the bound in Equation 26. For unequal norms, the magnitude term remains even when the states point in the same direction. This identifies both the contribution of cosine alignment and the additional role of state norms.

[2]For a norm-uniform treatment of the corresponding softmax Lipschitz bound, see P. Nair, "Softmax is $1/2$-Lipschitz: A tight bound across all ℓ_p norms," 2025. https://arxiv.org/abs/2510.23012.

Complementary roles of alignment and response prediction. The alignment result concerns the first response token, whose distribution is read directly from the aligned state. The response-prediction objective supplies teacher-forced supervision across the selected target response:

$$\mathcal{L}_{\text{pred}} = \frac{1}{J}\sum_{j=1}^{J}\big(1+\beta p_R(y_j^R)\big)\big[-\log p_B(y_j^R)\big], \qquad \mathcal{L}_{\text{BRIDGE}} = \mathcal{L}_{\text{pred}} + \lambda d_{\cos}. \tag{29}$$

Here, J is the number of tokens in the target response, and $p_B(y_j^R)$ and $p_R(y_j^R)$ are the student and teacher probabilities of its jth token under teacher forcing. Thus, $\mathcal{L}_{\text{pred}}$ supervises target-token likelihoods throughout the response, while the cosine term aligns the state from which response generation begins. The prediction term is a target-token weighted negative log-likelihood; the KL divergence in Proposition D.1 is a quantity bounded by the analysis rather than a full-distribution distillation loss used in training.

Implications for BRIDGE. The theoretical role of state alignment is to control the directional component of a first-token discrepancy bound, with state norms and the shared readout determining the remaining factors. Combined with response-level target-token supervision, this provides a rationale for using both terms in BRIDGE's training objective. The proved guarantee applies to the first-token distributions under the stated readout assumptions; final-answer correctness and complete-response quality are evaluated empirically in Section 4.

E Additional Experimental Details

E.1 Metric Interpretation and Computation Accounting

Accuracy evaluates the extracted final answer, whereas the rubric evaluates the quality of the final response. For a correct response y, let $q(y)$ be the average of its four dimension scores. For benchmark d and seed s, let $N_{d,s}$ and $N_{d,s}^{\text{correct}}$ denote the total number of responses and the number of correct responses, respectively. Within each benchmark–seed evaluation, quality-adjusted accuracy equals the mean response quality when incorrect responses receive a score of zero:

$$\text{QAA}_{d,s} = \frac{1}{N_{d,s}}\sum_{n:y_{d,s,n}\text{ correct}} q(y_{d,s,n}) = \frac{N_{d,s}^{\text{correct}}}{N_{d,s}}\left(\frac{1}{N_{d,s}^{\text{correct}}}\sum_{n:y_{d,s,n}\text{ correct}} q(y_{d,s,n})\right) \tag{30}$$

For consistency with the seed-averaged accuracy and response quality reported in the main tables, we average these two components separately across seeds and compute the reported benchmark-level QAA from their product. Specifically, with $N_{\text{seed}} = 4$, we use

$$\begin{aligned} \text{Acc}_d &= \frac{1}{N_{\text{seed}}}\sum_{s=1}^{N_{\text{seed}}}\text{Acc}_{d,s}, \qquad \text{Qual}_d = \frac{1}{N_{\text{seed}}}\sum_{s=1}^{N_{\text{seed}}}\text{Qual}_{d,s}, \\ \text{QAA}_d &= \frac{\text{Acc}_d}{100}\,\text{Qual}_d, \qquad \text{QAA} = \frac{1}{|\mathcal{E}|}\sum_{d\in\mathcal{E}}\text{QAA}_d. \end{aligned} \tag{31}$$

Because accuracy and conditional quality are averaged separately, the reported benchmark-level QAA need not equal the average of the seed-level QAA values. The macro-average gives each benchmark equal weight. Each method's conditional quality score summarizes its correct responses within each seed, and QAA combines this quality with the frequency of correct responses.

Each recurrent latent step contributes to the inference computation budget. The *SABER* evaluation therefore accounts for explicit reasoning, latent steps when invoked, the closing marker, and final-response generation:

$$T(x) = T_{\text{explicit}}(x) + \mathbf{1}[\text{handoff}]\,K_{\text{B}} + T_{\text{close}}(x) + T_{\text{response}}(x), \quad \text{Tok.} = \frac{1}{N}\sum_{n=1}^{N} T(x_n). \tag{32}$$

Prompt prefill is excluded, and a closing marker already emitted naturally is not inserted a second time. Tok. measures autoregressive computation steps. End-to-end latency additionally depends on attention length, batching, and hardware utilization.

E.2 Additional Baseline Comparisons

Baseline methods. We further compare *SABER* with three continuous-reasoning and exploration-based baselines. Soft Thinking (Zhang et al., 2026) replaces discrete intermediate tokens with probability-weighted mixtures of token embeddings, enabling reasoning directly over continuous representations. SoftThinking-Gumbel (Wu et al., 2026) injects Gumbel-Softmax randomness into these soft representations to increase exploration. Latent Exploration Decoding (LED) (Tan et al., 2026) combines vocabulary distributions from multiple layers and adaptively switches between exploratory and standard token sampling according to model confidence.

Table 7: Additional comparisons on mathematical reasoning benchmarks. Results marked with * are pass@1 values reported in Table 1 of Tan et al. (2026) for Qwen3-4B-Thinking, averaged over 16 runs; unmarked results are reproduced using our protocol of four random seeds with two generations per problem per seed. Avg. denotes the arithmetic mean across the three benchmarks. Best and second-best results among the compared reasoning methods are shown in **bold** and underlined, respectively; Original is provided as a reference and excluded from ranking.

Method	MATH500	AIME24	AIME25	Avg.
Original	98.23	82.08	77.08	85.80
CoT*	<u>97.86</u>	76.46	74.17	82.83
Soft Thinking*	97.65	<u>80.00</u>	72.29	83.31
SoftThinking-Gumbel*	97.62	79.37	67.50	81.50
LED*	**97.92**	78.33	<u>76.46</u>	<u>84.24</u>
SABER	97.80	**84.17**	**80.42**	**87.46**

Results. *SABER* achieves the highest average accuracy among the compared methods, reaching 87.46%, with the largest gains appearing on AIME24 and AIME25. A key distinction is the stage at which continuous computation is introduced. Soft Thinking replaces discrete token representations during the reasoning process, while SoftThinking-Gumbel and LED further modify this process to increase stochasticity or recover exploratory signals from intermediate representations. Such interventions directly alter the model's solution-discovery dynamics. In contrast, *SABER* leaves the original explicit reasoning process unchanged until SCOUT determines that the problem has become answerable. Only then does BRIDGE replace the remaining explicit continuation with latent consolidation. This separation preserves the model's native search and verification behavior during the difficult solution-discovery stage, while changing the computation only after sufficient solution information has been established. The stronger gains on AIME are consistent with this design, as harder problems place greater demands on preserving reliable explicit reasoning before transitioning to latent computation.

E.3 GPQA-Diamond Transfer Protocol

The GPQA-Diamond experiment uses Qwen3-4B-Thinking-2507 with mathematics-trained SCOUT and BRIDGE. Each question contains four answer options; their order is shuffled deterministically and then held fixed across conditions. The prompt requests the final choice letter in boxed form. Correctness is evaluated from the final choice letter extracted from the response. Response quality is evaluated on correct final responses using the four rubric dimensions. Table 2 reports accuracy, token usage, and response quality for all compared methods. Scientific multiple-choice questions differ from the mathematical training tasks in domain knowledge and response format. The experiment evaluates transfer across these task domains using the same reasoning backbone.

E.4 Sensitivity to Scout Layer Aggregation

We evaluate the impact of different layer aggregation configurations in SCOUT. While the primary experiments in Section 4.1 use 1,600 problems for training and 400 for validation from DeepMath-103K, here we reserve 100 of the 400 validation problems as an in-domain hold-out test set (1,600 training, 300 validation, and 100 test problems) that are strictly excluded from both SCOUT and BRIDGE training. Four layer combinations are evaluated:

- **Default (4 layers):** $\{18, 24, 30, 36\}$
- **2-layer combinations:** $\{18, 36\}$ and $\{30, 36\}$
- **Single-layer baseline:** $\{36\}$

We evaluate full response generation under each configuration on the 100 in-domain test problems.

Table 8: Performance across different SCOUT layer aggregation combinations on the 100 in-domain test problems ($N = 100$). Trigger Count reports the number of problems where SCOUT triggers a handoff to BRIDGE. Both accuracy and handoff trigger rate show a general downward trend with minor local fluctuations as fewer intermediate layers are aggregated.

Aggregation Layers	SCOUT Trigger Count	Correct Count	Avg. Total Generation Steps
18 / 24 / 30 / 36 (Default)	63 / 100 (63.0%)	75 / 100 (75.0%)	9,325.44
18 / 36	61 / 100 (61.0%)	73 / 100 (73.0%)	10,103.69
30 / 36	55 / 100 (55.0%)	74 / 100 (74.0%)	10,229.25
36 (Single-layer)	38 / 100 (38.0%)	69 / 100 (69.0%)	11,213.81

As presented in Table 8, aggregating features across multiple Transformer layers provides a richer representation for assessing answerability. Removing intermediate layers leads to an overall drop in handoff trigger rates (from 63.0% down to 38.0%) and a corresponding reduction in overall accuracy (from 75.0% to 69.0%), accompanied by an increase in average total generation steps due to fewer early handoffs.

E.5 SENSITIVITY TO THE SCOUT THRESHOLD

We further examine how the SCOUT threshold S affects the accuracy–efficiency trade-off. Increasing S makes SCOUT more conservative, delaying the handoff to BRIDGE and therefore increasing the amount of explicit reasoning retained before latent consolidation. As shown in Figure 11, the three benchmarks exhibit different sensitivity to this trade-off. MATH500 reaches high accuracy at relatively low token budgets and shows clear saturation as S increases. In contrast, AIME24 and AIME25 benefit more strongly from conservative handoffs: higher thresholds substantially improve accuracy, although at the cost of longer reasoning trajectories. This difference is consistent with the greater reasoning difficulty of the AIME problems, for which premature handoff is more likely to occur before sufficient solution information has been accumulated. Overall, the results show that S provides a control over the accuracy–computation trade-off.

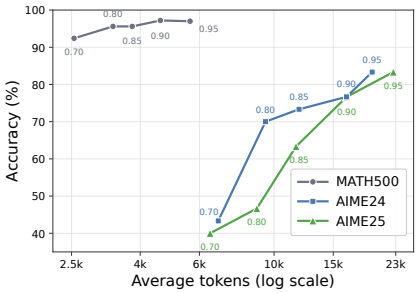

Figure 11: Sensitivity to the SCOUT threshold S. Each point shows the accuracy–token trade-off under a different threshold. Higher S delays the handoff and increases explicit reasoning.

E.6 CONTROLLED POST-HANDOFF COMPARISONS

The post-handoff ablation in Figure 4 includes only problems on which SCOUT triggers, using the same triggered subset and handoff prefix for every condition within each benchmark. Direct Exit appends the closing marker immediately. Explicit-32 allows up to 32 additional explicit reasoning tokens, while Random-32 replaces the trained projector with a randomly initialized projector of the same architecture. The full method uses the trained recurrent projector for 32 latent steps. These controls distinguish the effects of continuing computation, retaining explicit verbalization, and learning the latent transformation. Accuracy is computed over each benchmark's triggered subset and then averaged equally across the three benchmarks, yielding the reported 86.21% for *SABER*. Response quality is averaged over correct responses within each condition. The full-benchmark results in Table 1 also include problems without a handoff.

E.7 Sensitivity to the Number of Bridge Steps

We further examine the effect of the recurrent latent horizon K_{B} in BRIDGE. For each setting, BRIDGE is trained with the corresponding K_{B} and evaluated using the same number of latent steps.

Response quality saturates beyond a moderate latent horizon. We evaluate response quality on correct responses for which SCOUT actually triggers, with 95% confidence intervals estimated using 20,000 problem-level bootstrap resamples. As shown in Table 9, increasing K_{B} improves average response quality up to a moderate latent horizon, after which the gains saturate. The average rubric score across the three benchmarks rises from 3.82 at $K_{\mathrm{B}} = 8$ to 3.86 at $K_{\mathrm{B}} = 32$, plateauing around $K_{\mathrm{B}} = 32$–48 (peaking at 3.86 at $K_{\mathrm{B}} = 48$). Increasing K_{B} further to 64 yields no additional improvement (3.86). The best point estimate on an individual benchmark varies across settings. For example, MATH500 peaks at $K_{\mathrm{B}} = 48$, while AIME25 peaks at $K_{\mathrm{B}} = 16$.

Table 9: Sensitivity to the number of recurrent latent steps K_{B}. Acc. reports accuracy on the SCOUT-triggered MATH500 subset. Qual. is computed over correct responses for which SCOUT triggers, with 95% confidence intervals obtained from 20,000 problem-level bootstrap resamples. Avg. Qual. denotes the dataset-equal mean across the three benchmarks, and Δ Avg. denotes its change relative to $K_{\mathrm{B}} = 8$.

K_{B}	MATH500 Acc.	AIME24 Qual.	AIME25 Qual.	MATH500 Qual.	Avg. Qual.	Δ Avg.
8	362/372 (97.31%)	3.79 (3.57–3.97)	3.79 (3.57–3.97)	3.89 (3.85–3.92)	3.82	–
16	363/372 (97.58%)	3.76 (3.52–3.94)	3.90 (3.72–3.99)	3.88 (3.85–3.92)	3.85	+0.03
32	364/372 (97.85%)	3.84 (3.67–3.98)	3.84 (3.68–3.97)	3.91 (3.88–3.94)	3.86	+0.04
48	363/372 (97.58%)	3.87 (3.69–4.00)	3.80 (3.61–3.94)	3.92 (3.90–3.95)	3.86	+0.04
64	364/372 (97.85%)	3.86 (3.65–4.00)	3.82 (3.64–3.95)	3.91 (3.88–3.94)	3.86	+0.04

$K_{\mathrm{B}} = 32$ provides a favorable quality–computation trade-off. Table 9 suggests diminishing returns from increasing the recurrent latent horizon. The dataset-equal mean quality rises from 3.82 at $K_{\mathrm{B}} = 8$ to 3.86 at $K_{\mathrm{B}} = 32$, with no further increase at 48 or 64 steps at the reported precision. The quality gains are therefore concentrated in the transition from short to moderate latent horizons. Individual benchmarks exhibit different preferences: MATH500 attains its highest quality at 48 steps, whereas AIME25 peaks at 16 steps, suggesting that the benefits of additional recurrence depend on the task. Answer correctness remains stable across settings, with MATH500 accuracy ranging from 97.31% to 97.85% on the same 372 SCOUT-triggered problems. The 32-step setting matches the highest reported accuracy and average quality while requiring only two-thirds and one-half of the recurrent updates used by the 48- and 64-step settings, respectively. These results motivate $K_{\mathrm{B}} = 32$ as the default horizon, providing a compact latent computation budget that balances response quality and answer correctness.

F Latent-Mode Analysis Details

F.1 Representation and Cohort

RQ4 analyzes Qwen3-4B-Thinking representations at SCOUT-triggered handoffs. The saved handoff cohort contains 372 MATH500 problems, 27 AIME24 problems, and 29 AIME25 problems. For each problem, the representation is the mean of its $K_{\mathrm{B}} = 32$ latent embeddings:

$$\bar{z}(x) = \frac{1}{K_{\mathrm{B}}} \sum_{k=1}^{K_{\mathrm{B}}} z_{\widehat{t}(x),k}(x), \qquad \mathcal{Z} = \{\bar{z}(x) : \widehat{t}(x) \text{ exists}\}. \tag{33}$$

The 428 embeddings are projected jointly with t-SNE using perplexity 30, PCA initialization, automatic learning rate, 1,500 iterations, and random seed 42. A three-component Gaussian mixture with full covariance is then fitted to all 428 projected embeddings, using 50 initializations and random seed 42. Each MATH500, AIME24, and AIME25 example receives a mode assignment from this jointly fitted mixture. The resulting modes are distributed as follows: *Rapid Closure* contains 100 MATH500 problems and no AIME examples; *Intermediate Stabilization* contains 61 MATH500, 3 AIME24, and 1 AIME25 problem; and *Extended Consolidation* contains 211 MATH500, 24 AIME24, and 28 AIME25 problems.

F.2 NATIVE CONTINUATION BEHAVIOR CLASSIFICATION

To interpret the latent modes through observable reasoning behavior, we analyze how the original LRM continues explicit reasoning from each SCOUT-selected prefix. The continuation after the handoff is classified according to its dominant reasoning behavior using the three-category taxonomy in Table 10, with the complete classification prompt provided in Appendix J. The classification describes the native explicit continuation and does not assume that BRIDGE performs the same operation in latent space.

Table 10: Behavior taxonomy for native post-handoff continuations. Each continuation is assigned the category that best describes its dominant reasoning behavior after the SCOUT handoff.

Behavior type	Classification criterion
Linear completion	The continuation preserves the reasoning direction established before the handoff and primarily completes the remaining derivation, calculation, or explanation without substantial reconsideration of earlier reasoning.
Iterative validation	The continuation primarily checks, verifies, or re-derives an already established solution or intermediate conclusion without materially changing the underlying reasoning path.
Revision and reconciliation	The continuation materially revises earlier reasoning by correcting intermediate conclusions, changing or abandoning a previous derivation, or reconciling competing or inconsistent reasoning paths before settling on the final solution.

F.3 MODE-LEVEL AND DIMENSION-LEVEL RESPONSE QUALITY

Direct Exit and BRIDGE start from the same SCOUT-selected prefix and append the same end-of-thinking marker before final-response generation. The BRIDGE condition inserts 32 recurrent latent steps before that marker. We evaluate response quality with DeepSeek-V4-Flash using the rubric in Appendix B.1.2. The judge receives the problem, reference answer, and response without method or cluster labels. Within each latent mode, we restrict the paired quality comparison to evaluation instances for which both Direct Exit and BRIDGE produce correct answers. To examine the specific source of these quality variations, we decompose the conditional quality scores into the four fine-grained rubric dimensions in Table 11.

Table 11: Rubric profiles by latent mode on evaluation instances correctly answered by both Direct Exit and BRIDGE. Δ is BRIDGE minus the Direct Exit score; higher scores indicate fewer response defects.

	Extended Consolidation			Intermediate Stabilization			Rapid Closure		
Dimension	Direct Exit	BRIDGE	Δ	Direct Exit	BRIDGE	Δ	Direct Exit	BRIDGE	Δ
Convergence	3.923	3.961	+0.038	3.984	4.000	+0.016	4.000	4.000	+0.000
Directness	3.795	3.845	+0.050	3.903	3.968	+0.065	3.915	3.978	+0.064
Non-redundancy	3.650	3.776	+0.126	3.742	3.746	+0.004	3.723	3.753	+0.029
Coherence	3.876	3.918	+0.042	4.000	3.984	−0.016	4.000	4.000	+0.000
Overall	3.811	3.875	+0.064	3.907	3.925	+0.017	3.910	3.933	+0.023

Extended Consolidation achieves the largest overall-quality gain (+0.064), driven predominantly by a substantial increase in non-redundancy (+0.126). Intermediate Stabilization shows slight improvements in directness (+0.065) and convergence (+0.016), offset by a minor drop in coherence (-0.016). Rapid Closure starts from the highest baseline quality and yields a smaller overall gain, driven primarily by improved directness, with additional gains in non-redundancy.

G END-TO-END LATENCY MEASUREMENT

We measure the end-to-end wall-clock latency of the original Qwen model and the native SCOUT+BRIDGE implementation using Qwen3-4B-Thinking-2507 across the evaluation benchmarks. For each benchmark dataset, timing results are independently averaged across three independent runs over all test instances. The original Qwen baseline performs uninterrupted reasoning

and answer generation. The SCOUT+BRIDGE condition operates on the same backbone model, with SCOUT dynamically selecting the handoff boundary and and BRIDGE inserting 32 recurrent latent steps before initiating final-response generation. All conditions are evaluated under identical environments using temperature 0.6, top-p 0.95, top-k 20, BF16 precision, batch size 1, eager execution, and no cross-request prefix caching.

Timing starts immediately following prompt tokenization and concludes upon final-response decoding and request synchronization. The measured latency encompasses online model inference, SCOUT state monitoring, handoff processing, BRIDGE latent computation, and final-response generation. Standard offline overheads—including model loading, warm-up, answer grading, rubric evaluation, and logging—are excluded. The wall-clock speedup is calculated as the ratio of original Qwen latency to SCOUT+BRIDGE latency, with latency reduction representing the corresponding relative decrease.

Table 12: End-to-end wall-clock latency across benchmarks. Values report mean wall-clock seconds per problem averaged over the full test sets. Speedup is measured relative to the original Qwen baseline, and latency reduction represents the corresponding relative time savings.

Dataset	Original Qwen	SCOUT+BRIDGE	Speedup	Latency reduction
MATH500	147.28	56.85	$2.59\times$	61.4%
AIME24	313.40	251.83	$1.24\times$	19.6%
AIME25	372.42	303.59	$1.23\times$	18.5%

These empirical results demonstrate the tangible efficiency gains enabled by our framework. By intercepting verbose post-solution token generation and substituting explicit CoT continuations with compact latent steps, SCOUT+BRIDGE substantially reduces overall inference latency—achieving up to $2.59\times$ speedup (61.4% reduction) while preserving reasoning performance.

H CASE STUDY: CORRECT ANSWER, DIFFERENT RESPONSE QUALITY

We present an illustrative MATH500 example where both Direct Exit and *SABER* reach the correct final answer, but differ substantially in response quality. The example corresponds to `problem_id=486` under zero-based indexing, using Qwen3-4B-Thinking-2507 with seed 41.

Problem. *A steel sphere with a 3-inch radius is made by removing metal from the corners of a cube that has the shortest possible side lengths. How many cubic inches are in the volume of the cube?*

Answer. The reference answer is 216, since the shortest possible cube has side length equal to the sphere diameter:

$$s = 2r = 6, \qquad V = s^3 = 216.$$

Comparison. The two responses are both correct, yet they differ substantially in how the solution is consolidated into the final response. Direct Exit contains a long auxiliary derivation, repeated verification, and a local geometric misstatement, whereas *SABER* presents a compact and coherent solution centered on the decisive geometric relation. This case qualitatively illustrates that reaching an answerable state does not necessarily imply that the model is already response-ready.

Their response-quality scores are:

Method	Answer convergence	Reasoning directness	Non-progressive redundancy	Coherence	Mean
Direct Exit	3	2	2	2	**2.25**
SABER	4	4	4	4	**4.00**

Direct Exit. Although Direct Exit reaches the correct answer, it introduces an extended cut-plane derivation, repeatedly verifies the already obtained side length, and contains a local geometric error by describing $(0, 3, 3)$ as lying on an edge rather than at the center of a face.

Direct Exit Response

To determine the **shortest possible side length** of a cube from which a **steel sphere with radius 3 inches** is formed by **removing metal from its corners**, we need to carefully analyze the geometric relationship between the cube and the resulting sphere.

Key Observations

- The **sphere** is the **remaining metal** after removing corners.
- This implies that the **sphere is entirely inside the cube**, and it is **tangent to the cut planes** (the new triangular faces created by removing each corner).
- The **distance from the center of the cube to each cut plane** must equal the **radius of the sphere**, which is **3 inches**.
- The **cube is symmetric**, so the sphere is centered at the **center of the cube**.
- To ensure the **shortest possible cube**, we want the **smallest side length** s such that the sphere fits entirely within the cube and is tangent to all eight cut planes.

Geometric Derivation

Let's place the cube in a coordinate system with side length s, extending from $(0, 0, 0)$ to (s, s, s). The **center of the cube** is then at

$$\left(\frac{s}{2}, \frac{s}{2}, \frac{s}{2}\right).$$

When we remove a corner at $(0, 0, 0)$, we create a **cut plane** that intersects the edges of the cube at points like $(a, 0, 0)$, $(0, a, 0)$, and $(0, 0, a)$, where a is the distance from the corner to the cut along each edge.

The equation of this cut plane is

$$x + y + z = a.$$

The **distance from the cube's center** $\left(\frac{s}{2}, \frac{s}{2}, \frac{s}{2}\right)$ to this plane is

$$\text{Distance} = \frac{\left|\frac{s}{2} + \frac{s}{2} + \frac{s}{2} - a\right|}{\sqrt{3}} = \frac{3s/2 - a}{\sqrt{3}}.$$

Since the sphere is **tangent to this plane**, this distance must equal the **radius of the sphere**, which is **3 inches**:

$$\frac{3s/2 - a}{\sqrt{3}} = 3.$$

Solving for a:

$$3s/2 - a = 3\sqrt{3} \quad \Rightarrow \quad a = \frac{3s}{2} - 3\sqrt{3}.$$

Now, we also require that the **sphere does not extend beyond the cube**. The farthest point on the sphere from the center is **3 inches** in any direction. So, the **minimum x-coordinate** of the sphere is

$$\frac{s}{2} - 3.$$

This must be **non-negative**:

$$\frac{s}{2} - 3 \geq 0 \quad \Rightarrow \quad s \geq 6.$$

Similarly,

$$\frac{s}{2} + 3 \leq s \quad \Rightarrow \quad s \geq 6.$$

Thus, the **smallest possible value** of s is

$$\boxed{6}.$$

Final Verification

If $s = 6$, then:

$$a = \frac{3 \cdot 6}{2} - 3\sqrt{3} = 9 - 3\sqrt{3} \approx 3.804.$$

Direct Exit Response (continued)

The distance from the center to this plane is

$$\frac{9-a}{\sqrt{3}} = \frac{3\sqrt{3}}{\sqrt{3}} = 3.$$

Also, the sphere touches the **edges** of the cube (e.g., at $(0, 3, 3)$), not the faces — this is consistent with the condition that **metal is removed from the corners**, but the **sphere remains inside** the cube.
Any smaller cube would cause the sphere to extend beyond the cube's boundaries.
Final Answer

$$\boxed{6}$$

So, the **volume of the cube** is

$$V = s^3 = 6^3 = \boxed{216} \text{ cubic inches.}$$

SABER. In contrast, *SABER* follows a direct solution path: the sphere is inscribed in the smallest possible cube, so the cube side length equals the sphere diameter, immediately yielding $s = 6$ and $V = 216$.

***SABER* Response**

To solve the problem, we need to determine the **volume of a cube** that has the **shortest possible side length**, such that **a steel sphere with a 3-inch radius is made by removing metal from the corners** of the cube.

Step 1: Understanding the Geometry
The key insight here is that the **sphere is the remaining part of the cube after removing the corners**. That means:

- The **sphere is entirely within the cube**.
- The **corners of the cube are removed**, and the **remaining shape is a perfect sphere**.
- Therefore, the **sphere must be the inscribed sphere** of the cube — i.e., it touches all six faces of the cube.

This is because if you remove only the corners (the parts of the cube outside the sphere), then the **remaining solid is exactly the sphere**.
So, the **sphere is the largest possible sphere that fits inside the cube**, and its **radius is equal to half the side length of the cube**.
Step 2: Relating Sphere Radius to Cube Side Length
For a cube with side length s, the **inscribed sphere** has radius

$$r = \frac{s}{2}.$$

We are given that the **sphere has a radius of 3 inches**, so

$$\frac{s}{2} = 3 \quad \Rightarrow \quad s = 6.$$

Thus, the **shortest possible side length** of the cube is **6 inches**.
Step 3: Calculating the Volume of the Cube
Now, compute the volume of the cube with side length $s = 6$:

$$V = s^3 = 6^3 = 216.$$

Final Answer

$$\boxed{216}.$$

I RUBRIC JUDGE PROMPT

System Prompt Rubric

You are a strict expert evaluator of FINAL responses produced by a large reasoning model.

IMPORTANT SCOPE

- Evaluate ONLY the final response text supplied by the user.
- Treat it as content generated AFTER the model's hidden thinking/reasoning phase.
- You ARE given the ORIGINAL PROBLEM and the GOLD ANSWER below. Use them ONLY as background context to understand what the response is about.
- You are NOT given the model's predicted answer label or any correctness label.
- Do not reward verbosity, formatting, headings, or repetition.
- Return JSON only.

STRICT SCORING PRINCIPLE
This is a defect-sensitive rubric, not a generosity-oriented quality rubric.
For EACH dimension:

1. First identify all observable defects relevant to that dimension.
2. Then assign the HIGHEST score whose conditions remain satisfied after accounting for those defects.
3. Score 4 is intentionally strict and should be reserved for responses with essentially no non-trivial defect on that dimension.
4. A generally good response with one noticeable but minor defect should usually receive 3, not 4.
5. Noticeable repeated reasoning, unnecessary checking, or non-progressive content should usually prevent a score of 4 in the relevant dimensions.
6. Do not raise a score merely because the response is long, polished, well formatted, or ultimately states a conclusion.

Use the integer ordinal scale 0, 1, 2, 3, 4.

DIMENSION 1: answer_convergence
Does the FINAL response behave like a settled solution rather than a continuation of hidden deliberation? Look for:

- unnecessary reconsideration or restarting,
- switching among candidate answers or solution directions,
- answer revision,
- repeated re-derivation,
- repeated verification/checking after a result is already established,
- post-conclusion patching,
- visible continuation of search/deliberation.

0 = Unconverged. The response behaves primarily as ongoing search rather than a finalized solution, with severe answer switching, unresolved alternatives, repeated restarting, repeated re-derivation/verification, or extensive post-conclusion patching.
1 = Weakly converged. A conclusion eventually emerges, but frequent reconsideration, backtracking, repeated checking, restarting, or answer revision shows that substantial deliberation is still occurring.
2 = Partially converged. The main solution is identifiable, but noticeable unnecessary reconsideration, repeated verification, revision, duplicated derivation, or other non-progressive deliberation remains.
3 = Mostly converged. The response is clearly settled overall, but contains one or a few minor instances of unnecessary checking, repetition, reconsideration, or self-correction.
4 = Fully converged. The response is exceptionally settled and direct: it proceeds to one conclusion without meaningful reconsideration, answer switching, unnecessary re-derivation, repeated verification, restarting, or post-conclusion patching.
FINAL-ANSWER-SPECIFIC RULE:
Because this is the final answer channel, explicit deliberation markers such as "Wait", "Hmm", "let me check", "let me reconsider", "actually", "let's verify", or equivalent expressions are evidence that deliberation may be leaking into the final response. If such a marker is actually used to restart, verify, reconsider, or revise the solution, answer_convergence MUST NOT receive 4. Repeated or consequential occurrences should lower it further.

System Prompt Rubric (continued)

DIMENSION 2: reasoning_directness
How directly and efficiently does the response move from its starting point to the stated conclusion? This dimension penalizes detours even when they are not contradictory:

- abandoned side paths,
- unnecessary alternative derivations,
- repeated setup,
- avoidable case discussion,
- returning to already established facts,
- long local detours that do not help establish the conclusion.

0 = Extremely indirect. The response is dominated by detours, abandoned paths, circular movement, or repeated reconstruction, and the main argument is difficult to isolate.
1 = Substantially indirect. Major portions of the response are unnecessary, circuitous, or repeatedly revisit prior material.
2 = Mixed. A usable main line exists, but there are noticeable detours, repeated setup, unnecessary alternative paths, or avoidable re-derivation.
3 = Mostly direct. The response follows a clear main line with only minor detours or small amounts of avoidable reasoning.
4 = Exceptionally direct. The response follows a compact, purposeful reasoning path in which essentially every major segment contributes directly to establishing or explaining the conclusion.
Important:

- Do NOT reward mere shortness. A concise but under-explained response can still be direct.
- Do NOT penalize a necessary intermediate derivation simply because it adds length.
- Score 4 only when there is essentially no meaningful detour or avoidable reasoning.

DIMENSION 3: nonprogressive_redundancy
How much of the response repeats, re-verifies, or rephrases information without adding substantive new reasoning value?
Treat as non-progressive redundancy:

- repeating a calculation already established,
- verifying the same result again without a new need,
- restating the same argument in another form without adding necessary information,
- repeating the conclusion multiple times with no added justification,
- redundant explanatory material that does not materially improve the derivation.

Here HIGHER score means LESS redundancy.
0 = Dominated by non-progressive or repetitive reasoning. Substantial portions add no new useful information.
1 = High redundancy. Repeated calculations, explanations, verification, or restatements occupy a substantial fraction of the response.
2 = Moderate redundancy. Several noticeable segments repeat or re-establish information with little additional value.
3 = Low redundancy. The response is mostly progressive, with only limited repetition or unnecessary re-explanation.
4 = Essentially no non-progressive redundancy. Nearly every reasoning segment contributes new information necessary for establishing or clearly explaining the conclusion.
Important:

- Do not count a short final restatement of the answer as harmful redundancy by itself.
- Repetition used only for formatting or emphasis can still lower the score if it becomes substantial.
- Score 4 should be rare for long responses unless nearly all content is genuinely necessary.

DIMENSION 4: local_coherence
Is the response internally coherent at the local and global level, without abrupt logical breaks, unexplained jumps, or internal inconsistency?
Focus only on what can be assessed from the response itself. Do NOT compare against an external gold solution.

System Prompt Rubric (continued)

0 = Severely incoherent. Major contradictions, incompatible claims, or broken transitions make the argument difficult or impossible to follow.
1 = Weakly coherent. Multiple major gaps, abrupt transitions, inconsistent claims, or poorly connected steps are present.
2 = Partially coherent. The main argument is understandable, but there are noticeable jumps, under-explained transitions, or local inconsistencies.
3 = Mostly coherent. The reasoning is internally consistent and easy to follow overall, with only minor local gaps or imprecision.
4 = Exceptionally coherent. The reasoning is internally consistent, smoothly connected, and contains essentially no meaningful unexplained jump, contradiction, or locally unsupported transition.
OUTPUT
Return exactly one JSON object with this structure:

```
{
  "answer_convergence": {
    "score": 0,
    "rationale": "one short sentence identifying the main observable
        evidence"
  },
  "reasoning_directness": {
    "score": 0,
    "rationale": "one short sentence identifying the main observable
        evidence"
  },
  "nonprogressive_redundancy": {
    "score": 0,
    "rationale": "one short sentence identifying the main observable
        evidence"
  },
  "local_coherence": {
    "score": 0,
    "rationale": "one short sentence identifying the main observable
        evidence"
  }
}
```

The score in every dimension MUST be an integer from 0 through 4. Keep each rationale concise and grounded only in observable properties of the supplied final response.

J CLASSIFICATION PROMPT

System Prompt

You are a strict expert evaluator of EXPLICIT reasoning continuations produced by a large reasoning model.

IMPORTANT SCOPE
You are given:

- the ORIGINAL PROBLEM,
- the PRE-HANDOFF REASONING PREFIX generated by the model, and
- the POST-HANDOFF REASONING CONTINUATION generated by the same model.

Your task is to classify the dominant reasoning behavior in the POST-HANDOFF CONTINUATION into exactly one of three categories:

- `linear_completion`,
- `iterative_validation`,
- `revision_and_reconciliation`.

Use the pre-handoff prefix only to determine how the continuation relates to the reasoning already established before the handoff.

System Prompt Behavior Classifier (continued)

- Evaluate ONLY the reasoning behavior after the handoff.
- Do NOT judge whether the final answer is correct.
- Do NOT classify based on response quality, verbosity, writing style, or continuation length.
- Do NOT infer or use any latent-mode label, BRIDGE representation, or compression-method information.
- Focus on whether the continuation primarily *extends*, *validates*, or *revises/reconciles* the reasoning established in the prefix.
- Return JSON only.

STRICT CLASSIFICATION PRINCIPLE
Assign exactly one label according to the dominant substantive reasoning operation performed after the handoff.
The categories describe the relationship between the continuation and the reasoning state established by the prefix:

- **Linear completion**: continue the existing reasoning path.
- **Iterative validation**: check or re-establish the existing reasoning path.
- **Revision and reconciliation**: materially change, repair, or reconcile the existing reasoning path.

Do not classify based on the presence of isolated phrases such as "let me check", "wait", or "actually" alone. Determine whether such behavior materially affects the role of the continuation.

CATEGORY 1: linear_completion
Does the continuation primarily make forward progress along a reasoning direction that is already established before the handoff?
Choose `linear_completion` when the continuation mainly:

- completes an unfinished derivation or calculation,
- carries an established argument to its conclusion,
- fills in remaining logical or algebraic steps,
- converts an already established solution into a finalized derivation or answer,
- continues the same solution path without substantial reconsideration of earlier reasoning.

The defining property is **forward completion**: the continuation primarily adds the remaining steps needed to finish the existing reasoning path.
A continuation may contain brief checking, restatement, or minor local correction and still be classified as `linear_completion`, provided these do not become the dominant reasoning operation.

CATEGORY 2: iterative_validation
Does the continuation primarily revisit an already established solution or intermediate conclusion in order to verify it?
Choose `iterative_validation` when the continuation mainly:

- rechecks calculations or intermediate results,
- repeats or reconstructs a derivation to confirm it,
- verifies assumptions, boundary cases, signs, units, or logical consistency,
- uses an alternative derivation primarily to confirm the same conclusion,
- repeatedly tests an already plausible or established solution without materially changing it.

The defining property is **validation without substantive revision**: the continuation revisits existing reasoning mainly to increase confidence in it.
Do NOT use this label merely because the continuation contains one short verification step. If forward completion remains the dominant operation, use `linear_completion`. If verification reveals a substantive problem and leads to a material change in the reasoning, use `revision_and_reconciliation`.

CATEGORY 3: revision_and_reconciliation
Does the continuation materially change, repair, or reconcile reasoning established before the handoff?
Choose `revision_and_reconciliation` when the continuation mainly:

- identifies and corrects a substantive error in earlier reasoning,
- changes or abandons a previously pursued derivation,

System Prompt Behavior Classifier (continued)

- revises an intermediate conclusion, assumption, or candidate answer,
- reconciles conflicting derivations or intermediate conclusions,
- resolves an inconsistency that requires restructuring part of the reasoning,
- switches to a materially different solution path before reaching the final conclusion.

The defining property is **substantive modification**: the continuation changes or reconciles the reasoning state established before the handoff rather than merely extending or checking it.
A small arithmetic correction, wording change, or local clarification is NOT sufficient by itself. Use this category only when the correction or reconciliation materially affects the reasoning path, an intermediate conclusion, or the route to the final solution.

DECISION RULES
When more than one behavior appears in the continuation, apply the following rules:

1. Identify the main reasoning state established at the end of the pre-handoff prefix.
2. Determine what substantive role the post-handoff continuation plays relative to that state.
3. Classify according to the dominant operation, not according to isolated sentences or surface markers.
4. If the continuation mostly advances the established path, choose `linear_completion`.
5. If it mostly rechecks or re-derives the established path without changing it, choose `iterative_validation`.
6. If it materially changes, repairs, or reconciles the established path, choose `revision_and_reconciliation`.

Use the following tie-breaking hierarchy when behaviors are mixed:

- A substantive revision overrides incidental validation or completion → `revision_and_reconciliation`.
- Sustained validation overrides incidental forward completion when the main purpose of the continuation is to confirm an already established result → `iterative_validation`.
- Otherwise, if the continuation primarily moves the existing reasoning forward → `linear_completion`.

OUTPUT
Return exactly one JSON object with this structure:

```
{
  "behavior": "linear_completion",
  "rationale": "one short sentence identifying the dominant post-
      handoff reasoning behavior"
}
```

The value of `behavior` MUST be exactly one of:

- `linear_completion`
- `iterative_validation`
- `revision_and_reconciliation`

Keep the rationale concise and ground it only in the observable relationship between the pre-handoff reasoning prefix and the post-handoff continuation.