SPIKE-TRIGGERED ADAPTIVE REASONING STEERING TO MITIGATE COGNITIVE INERTIA IN LARGE REASONING MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

While Large Reasoning Models (LRMs) have achieved remarkable performance by scaling test-time compute, they frequently suffer from *cognitive inertia*, a failure pattern manifesting in two distinct modes: 1) *overthinking*, where models spiral into redundant verification loops long after a solution is reached, and 2) *reasoning rigidity*, where models stubbornly adhere to incorrect parametric priors even when they contradict explicit task constraints. Previous inference-time methods designed to mitigate these reasoning failures primarily focus on overthinking via suppressive early-exit strategies. However, by relying on surface-level textual heuristics that serve as unreliable proxies for reasoning quality, these methods often prematurely terminate potentially recoverable reasoning paths. Furthermore, they remain fundamentally blind to reasoning rigidity, as they lack any mechanism to redirect a model that has confidently committed to an incorrect trajectory. To address these limitations, we propose **STARS (Spike-Triggered Adaptive Reasoning Steering)**, a training-free framework designed to actively correct reasoning trajectories by monitoring internal latent dynamics. STARS identifies *cognitive pivots*—critical moments of reasoning transition—by detecting distinct L_2 distance spikes in the hidden states. Upon detection, the framework employs geometric trajectory analysis to diagnose the structural nature of the transition and injects state-aware language cues to steer the model on-the-fly. Our experiments across diverse benchmarks show that STARS consistently improves accuracy—by up to 6.6% on AIME25 and 6.5% on ConditionedMath—while consistently lowering end-to-end latency relative to existing baselines. By resolving internal conflicts rather than merely truncating output, STARS ensures a superior accuracy-compute trade-off, effectively rescuing failed reasoning paths.

1 INTRODUCTION

The paradigm of Large Language Models (LLMs) has recently shifted from parameter scaling to scaling test-time compute. Models such as o1 (Jaech et al., 2024) and DeepSeek-R1 (Guo et al., 2025) demonstrate that extending reasoning chains can yield performance gains comparable to massive parameter scaling (Muennighoff et al., 2025).

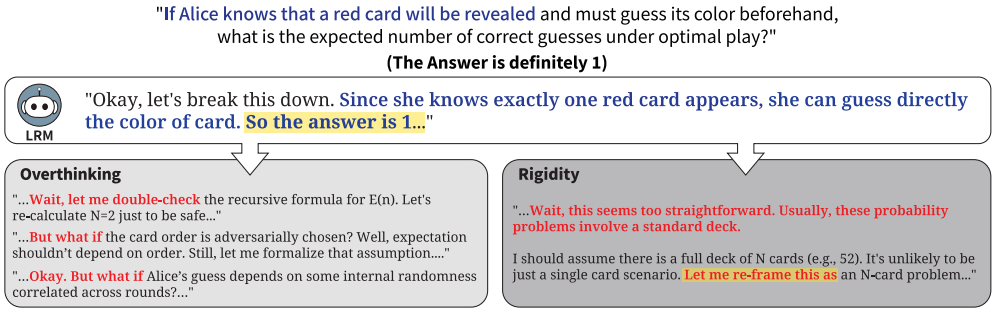


Figure 1: **Cognitive inertia in LLMs.** An illustration of two failure modes—*overthinking* (inertia of motion) and *reasoning rigidity* (inertia of direction).

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.

However, extended generation introduces *cognitive inertia*—a failure of adaptive reasoning where a model's trajectory becomes self-reinforcing and resists mid-course correction (Peng et al., 2025). As models are encouraged to "think longer", two distinct failure modes emerge as illustrated in Figure 1. First, models may persist in *overthinking*, spiraling into redundant verification loops (e.g., repeatedly double-checking converged formulas) long after a solution is reached (Chen et al., 2025b; Ghosal et al., 2025). Second, they exhibit *reasoning rigidity*, where strong parametric priors override explicit task constraints—such as forcing a familiar 52-card interpretation despite being told only a single card exists (Wu et al., 2024; Jang et al., 2025). Despite their surface differences, both modes share a common underlying mechanism: a self-reinforcing trajectory that resists correction even when contradicting task constraints. Drawing on the psychological concept of *cognitive inertia* (McGuire, 1960), we unify these behaviors under the respective labels of *overthinking* (inertia of motion) and *reasoning rigidity* (inertia of direction). This framing suggests that simply "thinking longer" does not guarantee better outcomes; instead, it necessitates a mechanism that actively monitors and corrects the reasoning trajectory itself.

To address these failures, recent works have proposed various inference-time control mechanisms. Prompt-based approaches (Han et al., 2025) attempt to guide reasoning through input-level instructions, yet remain static and are frequently overridden by parametric priors. Dynamic approaches, such as DEER (Yang et al., 2025b) and ConCISE (Qiao et al., 2025), focus on overthinking by using token-level entropy or confidence signals for early-exit strategies to truncate generation. However, by relying on surface-level textual heuristics, these methods often prematurely terminate potentially recoverable reasoning paths (Wang et al., 2025). More critically, these suppressive approaches remain blind to reasoning rigidity: a model confidently committed to an incorrect trajectory does not need to be stopped—it needs to be redirected. Consequently, no existing inference-time method simultaneously addresses both failure modes, leaving self-reinforcing trajectories unresolved.

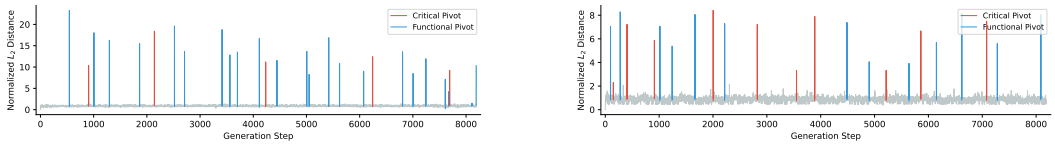
To resolve the limitations of surface-level monitoring in regulating cognitive inertia, we propose a shift toward analyzing the internal latent dynamics of LRMs. Through empirical analysis of hidden state trajectories, we discover that the signals required for such regulation are rooted in specific functional boundaries we term *cognitive pivots*, where the model undergoes a significant reconfiguration in its reasoning mode. We consistently observe that these pivotal moments are marked by *hidden state spikes*—sudden, high-magnitude displacements in the hidden states that appear across models and benchmarks. Crucially, these spikes are fundamentally neutral, representing a state of elevated plasticity rather than an inherent indicator of error. To differentiate between these distinct outcomes, we conceptually distinguish between routine *functional pivots* and pathological *critical pivots*, establishing a systematic basis for diagnosing internal conflicts and determining precisely when intervention is required to rectify reasoning trajectories.

Building on this, we introduce **STARS (Spike-Triggered Adaptive Reasoning Steering)**, a training-free framework employing a three-stage cycle: **Detect**, **Select**, and **Steer**. STARS detects latent spikes, diagnoses whether they indicate *critical pivots* via geometric analysis, and steers the reasoning path on-the-fly using state-aware language cues. A *shifting suffix* redirects the model away from rigid priors by grounding it in the problem context, while a *loop breaker suffix* disrupts redundant cycles to consolidate progress toward a final answer. This closed-loop approach allows LRMs to navigate complex trajectories without additional fine-tuning.

We evaluate STARS on diverse benchmarks, including instability-oriented diagnostics (Jang et al., 2025; Wu et al., 2024) and high-complexity problems like AIME (MAA Committees). STARS consistently improves accuracy—by up to 6.6% on AIME25 and 6.5% on ConditionedMath—by correcting erroneous trajectories before they entrench. Unlike suppressive baselines, STARS achieves a superior accuracy-compute trade-off, where token reduction is a natural byproduct of trajectory correction rather than an explicit target. Overall, STARS provides a robust, training-free mechanism to optimize LRM reasoning, consistently lowering end-to-end latency while effectively rescuing failed reasoning paths.

2 The hidden state spike: A window into cognitive inertia

Inspired by the psychological concept of cognitive inertia (McGuire, 1960), we term the observed tendency toward structural persistence in LRMs as *cognitive inertia*—a failure of adaptive reasoning where the model's internal representation develops a self-reinforcing trajectory. As models scale



(a) **DeepSeek-7B** (Layer 13) (b) **Qwen3-4B** (Layer 2)

Figure 2: **Latent L_2 spikes across models.** Normalized hidden state displacements for a shared problem instance. The varying distribution of **Functional** and **Critical** pivots reflects heterogeneous reasoning dynamics across different models.

test-time compute through extended chains of thought, their internal states can become resistant to mid-course corrections, even when the context provides clear evidence for a pivot or termination. This phenomenon represents a departure from ideal inference, where the model's trajectory becomes decoupled from the explicit task constraints. To provide a systematic framework for analyzing these failures, we unify these established behaviors under the conceptual umbrella of cognitive inertia, which manifests through two primary modes:

- **Overthinking** (*inertia of motion*): Similar to an object that resists braking, the model persists in redundant verification loops despite having reached a plausible solution, continuing to generate without meaningful reasoning progress (Ghosal et al., 2025).
- **Reasoning Rigidity** (*inertia of direction*): Similar to an object that resists changing course, the model commits to an incorrect trajectory driven by dominant parametric priors, overriding task-specific constraints in favor of familiar, canonical templates (Jang et al., 2025).

2.1 Cognitive Pivots: The Locus of Reasoning Transition

In both overthinking and reasoning rigidity, failure patterns tend to crystallize at specific junctions of reasoning reconfiguration rather than accumulating gradually. These discrete transition points, which we term *cognitive pivots*, mark functional boundaries—such as shifting from calculation to verification—where the reasoning mode is in flux. We hypothesize that at these boundaries, the latent trajectory undergoes a phase of elevated plasticity, making it particularly susceptible to the onset of inertial drift. To examine this hypothesis directly, we turn to the internal representation space, where these transitions should be visible as the trajectory begins to diverge.

2.2 The Hidden State Spike Phenomenon

To quantify the internal dynamics of reasoning, we monitor the layer-wise Euclidean distance of the hidden states in the decoder. We define the latent displacement $\delta_t^{(l)}$ at time step t and layer l as the magnitude of the state update:

$$\delta_t^{(l)} = \|h_t^{(l)} - h_{t-1}^{(l)}\|_2 \tag{1}$$

Figure 2 illustrates that $\delta_t^{(l)}$ exhibits distinct spike patterns that are clearly distinguishable from background generation noise across layers. We term these events *hidden state spikes*. This quantity is the instantaneous *velocity* of the reasoning trajectory in representation space, which Zhou et al. (2026) formalize within a differential-geometric framework and show to be controlled in both magnitude and direction by logical structure. A spike therefore identifies a step at which the internal state is reconfigured far more than during ordinary generation, rather than an arbitrarily chosen statistic. We hypothesize that these spikes characterize the precise transitions in the reasoning process, marking the temporal windows where the internal state is highly plastic and, consequently, vulnerable to the onset of cognitive inertia. As visualized in Figure 3, these latent fluctuations consistently align with the functional boundaries where the model's reasoning mode is in flux, providing a robust indicator for monitoring on-the-fly well before failures manifest in the generated text.

Spike-Aligned Context Windows

Critical Pivot

`Wait, maybe not. Wait, wait! Wait a second. Maybe the problem is in the` **`style`**

Functional Pivot

$(2 + 8 + 1)/4 = 1025/4$`, correct. Oh, nice! So the factors are integers in the` **`numer`**

Figure 3: Spike-aligned context windows. Red-highlighted tokens indicate detected latent spikes. Each window shows a short preceding context for interpretability.

2.3 Semantic taxonomy: functional vs. critical pivots

To characterize their semantic roles, we manually annotated 327 transitions across two reasoning domains and scaled the analysis to 2,068 spikes using an LLM judge (Appendix F.1). As illustrated in Figure 3, spikes align with pivotal reasoning transitions but are fundamentally neutral: on ConditionedMath, 72.8% are benign *functional pivots* (e.g., initiating a calculation), whereas 27.2% are *critical pivots* associated with cognitive inertia. This pattern remains consistent across domains and at a $\sim 6\times$ larger annotation scale, with critical pivots remaining a minority. Crucially, both types exhibit similarly high-magnitude latent updates, so spike magnitude alone cannot distinguish them. Nevertheless, latent spikes provide an early diagnostic window into reasoning transitions before their outcomes become entrenched in the generated trajectory. This necessitates a framework that treats spikes as transition signals and applies trajectory-level diagnosis before intervention.

3 STARS: Spike-triggered adaptive reasoning steering

We propose **STARS (Spike-Triggered Adaptive Reasoning Steering)**, a training-free framework designed to counteract cognitive inertia at inference time. Building on the insights established in Section 2, STARS operationalizes latent monitoring into an actionable three-stage cycle as described in Figure 4: (i) **Detect**, which identifies candidate pivots via normalized spikes; (ii) **Select**, which diagnoses trajectory geometry to resolve signal neutrality; and (iii) **Steer**, which injects state-aware natural language cues to recalibrate the reasoning path.

3.1 Stage 1: Detect - Identifying candidate spike

The Detect stage isolates meaningful reasoning transitions from background latent fluctuations. To turn these candidate spikes, which Section 2.3 characterizes as neutral transition signals, into a reliable trigger across architectures, we use a robust spike detection procedure that accounts for activation variance.

Robust spike signal extraction To ensure the framework is architecture-aware and scale-invariant, we monitor the normalized displacement signal $\mathcal{S}_t^{(l)}$ for each layer l at time step t:

$$\mathcal{S}_t^{(l)} = \|h_t^{(l)} - h_{t-1}^{(l)}\|_2 \,/\, \|h_{t-1}^{(l)}\|_2, \tag{2}$$

where $h_t^{(l)}$ denotes the hidden state at layer l.

Unlike naive thresholding, which is sensitive to outliers, we estimate an adaptive threshold $\tau^{(l)}$ using the **Median Absolute Deviation (MAD)**. This robust statistical measure allows us to define the set of significant pivot events $\mathcal{P}^{(l)}$ as those exceeding the baseline volatility:

$$\tau^{(l)} = \text{median}(\mathcal{S}^{(l)}) + k \cdot \text{MAD}(\mathcal{S}^{(l)}), \tag{3}$$

where k is a sensitivity parameter calibrated to maximize the detection of latent reconfigurations. We then isolate the significant reasoning transitions by defining the set of pivot events $\mathcal{P}^{(l)}$ as the collection of signals exceeding this robust threshold:

$$\mathcal{P}^{(l)} = \{s \in \mathcal{S}^{(l)} \mid s > \tau^{(l)}\}. \tag{4}$$

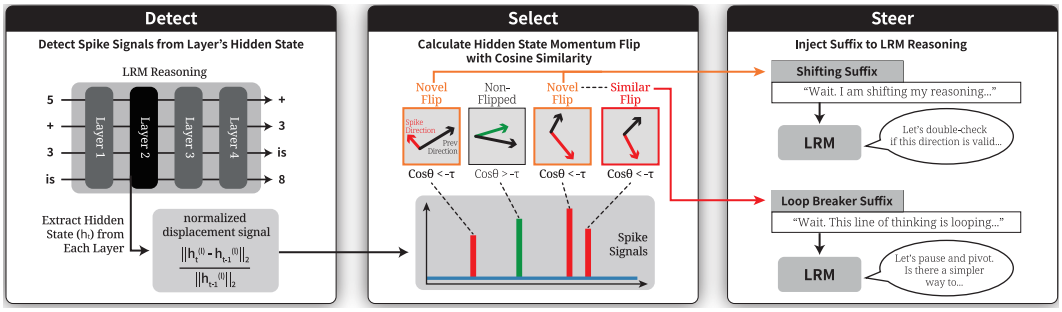


Figure 4: **The STARS three-stage steering cycle.** The framework (1) detects candidate pivots through L_2 spikes, (2) diagnoses their structural nature via geometric analysis, and (3) adaptively steers the trajectory using state-aware language suffixes.

This ensures that subsequent prominence metrics are grounded in structurally meaningful transitions rather than incidental numerical fluctuations.

Quantifying signal prominence Because reasoning transitions are more pronounced in specific functional layers, we introduce the **Spike Prominence Ratio (SPR)** to identify the optimal monitoring locus. Unlike a standard Signal-to-Noise Ratio (SNR), SPR utilizes median-based statistics on the set of detected candidates $\mathcal{P}^{(l)}$ to quantify how distinctly the reasoning pivots rise above the stochastic background:

$$\text{SPR}^{(l)} = \frac{\text{median}(\mathcal{P}^{(l)}) - \text{median}(\mathcal{S}^{(l)})}{\text{MAD}(\mathcal{S}^{(l)})} \tag{5}$$

This metric identifies the *reasoning hub* within the model architecture where the signal for cognitive pivots is most reliable.

Spike identification By selecting the representative layer l^* that maximizes $\text{SPR}^{(l)}$, we obtain a temporally aligned indicator of high-magnitude latent transitions:

$$\mathcal{I}_t = \mathbb{I}\left(\mathcal{S}_t^{(l^*)} > \tau^{(l^*)}\right) \tag{6}$$

This indicator $\mathcal{I}_t$ serves as the foundational trigger for the STARS framework. It identifies the precise *window of opportunity*—a state of elevated plasticity—enabling the subsequent diagnostic and steering phases to resolve the functional meaning of the transition as generation proceeds. Further details on layer selection and hyperparameter calibration are provided in Appendix E.

3.2 Stage 2: Select - Diagnosing Dynamics

The detection of a spike alone is insufficient for intervention, as it represents a fundamentally neutral state of plasticity. In this stage, STARS resolves this ambiguity by analyzing the geometry of the hidden-state trajectory to distinguish between benign transitions and pathological inertia.

Detecting directional flips We first test whether a spike corresponds to a conflict in reasoning direction by computing the cosine similarity $c_t = \text{CosSim}(v_t, v_{t-1})$ between consecutive update vectors $v_t = h_t - h_{t-1}$ and v_{t-1}. A sharp negative alignment ($c_t < -\tau_{\text{flip}}$) indicates a **Latent Flip**, reflecting a sudden reversal in the reasoning trajectory.

Diagnosing recurrence For detected directional flips, we further assess whether the instability reflects a novel deviation or a recurrent state. This latent view of looping is motivated by Duan et al. (2026), who show that self-reinforcing loops contract the representation geometry across repetition cycles, making a loop detectable in the hidden state before it surfaces as repeated text. We maintain a memory bank $\mathcal{H}_{\text{bank}}$ that stores hidden states at flip events and define the **Recurrence Score** as:

$$\rho_t = \max_{h_k \in \mathcal{H}_{\text{bank}}} \text{CosSim}(h_t, h_k). \tag{7}$$

The memory bank is updated only at flip-aligned moments, ensuring that recurrence is evaluated relative to prior unstable states. At the first flip of a sequence $\mathcal{H}_{\text{bank}}$ is empty; we define $\rho_t = 0$ in this case, so that the first flip is deterministically diagnosed as a novel deviation. Based on ρ_t, we distinguish between:

- **Novel Instability ($\rho_t \leq \tau_{\text{recur}}$):** The model is attempting a new direction, but with high conflict.
- **Cognitive Recurrence ($\rho_t > \tau_{\text{recur}}$):** The model has returned to a previous conflict state, indicating a reasoning loop.

3.3 Stage 3: Steer - adaptive cueing

Finally, STARS injects state-aware natural language suffixes conditioned on the diagnosed dynamics, prompting the model to recalibrate its path rather than enforcing blunt termination. These cues are structured to provide (i) *state acknowledgment*, surfacing the diagnosed condition, and (ii) *action guidance*, providing minimal bias for correction.

When a transition is identified as Novel Instability, we inject a **Shifting suffix** (e.g., "*Wait. I am shifting my reasoning...*") that explicitly acknowledges a sudden directional change in the reasoning trajectory and encourages the model to re-evaluate whether the new direction is grounded in the problem context.

In contrast, when the state is diagnosed as Cognitive Recurrence, we inject a **Loop breaker suffix** (e.g., "*Wait. This line of thinking is looping...*") that reflects the presence of a recurring reasoning state. By surfacing the repetitive pattern, the suffix biases the model away from inertial continuation and toward exploring an alternative reasoning trajectory. We provide additional details on suffix design and thresholds in Appendices E.3 and E.4.

4 Experiments

4.1 Experimental setup

Datasets We evaluate our framework on a collection of benchmarks covering complementary reasoning regimes. We use ConditionedMath (Jang et al., 2025) and MathPerturb (Huang et al., 2025a) to probe reasoning rigidity and perturbation-induced instability, and AIME24/AIME25 (MAA Committees) to assess generalization under long-horizon mathematical reasoning. We also use MATH500 (Hendrycks et al., 2021), a standard mathematical reasoning benchmark, to evaluate performance on general problem-solving. In addition, we evaluate domain generalization with PuzzleTrivial (Jang et al., 2025), which targets logical reasoning under disrupted intuitive priors, and GPQA-Diamond (Rein et al., 2024), which focuses on graduate-level scientific reasoning. Additional details on datasets are in Appendix D.1.

Models We evaluate our framework on three open-source models spanning distinct scales and post-training paradigms: Qwen3-4B-Thinking-2507 (Yang et al., 2025a) as a compact reasoning model, and DeepSeek-R1-Distill-Qwen-1.5B/7B (Guo et al., 2025) to examine the impact of distillation-based post-training on latent dynamics.

Baselines We benchmark STARS against representative inference-time intervention strategies. To ensure a controlled evaluation, all selected baselines are training-free and operate purely during the decoding phase. Specifically, to address inference-time redundancy, we include DEER (Yang et al., 2025b), an approach that employs dynamic early-exit decisions grounded in token-level confidence signals. We also evaluate the training-free decoding variant of ConCISE (Qiao et al., 2025), which guides generation using step-wise confidence signals. More details on baselines are provided in Appendix D.

Hyperparameters All detection and diagnostic thresholds in STARS are calibrated on held-out samples disjoint from evaluation benchmarks. We use MAD to determine model-specific spike thresholds $\tau^{(l^*)}$ for noise resistance, while diagnostic thresholds ($\tau_{\text{flip}}, \tau_{\text{recur}}$) are derived from distributional analyses of cosine signals to isolate structural transitions. Once calibrated, these parameters remain fixed across all benchmarks to ensure generalization. See Appendices E.1 and E.4 for details.

Table 1: **Main results.** Accuracy (%) across benchmarks, reported as mean ± std over 5 runs. *Abbreviations: M-Pert (MathPerturb), CondMath (ConditionedMath), Puzzle (PuzzleTrivial), GPQA-D (GPQA-Diamond).* Bold and underlined entries indicate the best and second-best results, respectively.

Method	M-Pert	CondMath	MATH500	AIME24	AIME25	Puzzle	GPQA-D	Avg.
DeepSeek-1.5B								
Vanilla	62.08±1.27	46.54±4.28	77.68±2.49	**28.00**±4.52	22.67±3.27	28.48±6.53	34.65±2.57	42.87
DEER	57.66±1.76	46.15±5.96	78.60±1.30	**28.00**±2.67	22.67±3.27	31.52±4.11	35.15±1.88	42.82
ConCISE	58.70±3.62	46.54±3.73	79.12±1.47	26.67±2.98	25.33±2.67	31.52±4.54	34.55±1.30	43.20
STARS	**62.60**±1.27	**53.08**±2.61	**79.36**±1.27	**28.00**±4.00	**27.33**±3.89	**37.58**±3.64	**35.45**±1.95	**46.20**
DeepSeek-7B								
Vanilla	82.34±3.14	48.08±5.01	89.52±0.41	53.33±2.11	39.33±1.33	40.00±2.27	47.47±1.72	57.15
DEER	74.03±1.16	45.77±1.88	86.92±1.01	49.33±6.80	40.67±3.89	37.58±5.62	46.06±3.02	54.34
ConCISE	81.04±2.80	46.54±3.08	89.60±0.76	47.33±6.46	39.33±4.42	41.82±4.02	**47.98**±2.28	56.23
STARS	**84.68**±0.97	**54.62**±1.96	**90.08**±0.79	**55.33**±3.40	**43.33**±2.98	**44.23**±3.09	47.78±1.30	**60.01**
Qwen3-4B								
Vanilla	94.55±2.08	56.54±1.54	94.16±0.32	76.00±3.89	70.67±5.33	73.33±3.53	**65.56**±1.25	75.83
DEER	90.65±1.72	56.92±1.96	93.44±0.38	76.00±4.90	72.00±5.99	74.55±3.64	62.93±1.04	75.21
ConCISE	93.77±1.91	54.23±4.45	93.96±0.36	75.33±3.40	74.67±4.52	72.73±2.71	62.22±1.29	75.27
STARS	**95.32**±1.04	**58.85**±3.12	**94.64**±0.65	**77.33**±3.89	**77.33**±3.89	**76.36**±3.53	64.34±1.62	**77.64**

4.2 Comparative analysis across reasoning regimes

Performance on failure-prone reasoning Table 1 summarizes performance across all benchmarks, where STARS achieves the highest average accuracy across all three model scales. On MathPerturb and ConditionedMath, where failures are typically driven by redundant verification loops or local indecision, STARS consistently improves accuracy on tasks highly susceptible to cognitive inertia. For example, on ConditionedMath (DeepSeek-7B), STARS improves accuracy by 6.54% while redirecting the model away from inertial drift without suppressing necessary intermediate reasoning.

Generalization and robustness STARS demonstrates consistent robustness across complementary evaluation settings beyond its primary targets. Evaluation on MATH500 confirms that spike-triggered steering maintains or marginally improves accuracy, indicating no interference with standard mathematical problem-solving. For long-horizon tasks such as AIME, where extended reasoning is essential, STARS avoids aggressive truncation and instead selectively intervenes at critical pivots, preserving the generation budget necessary for complex derivations. Notably, AIME24/25 were not curated to elicit cognitive inertia, yet STARS attains some of its largest gains there (up to 6.6% on AIME25). We attribute this to the longer reasoning chains AIME problems require, which naturally produce more opportunities for cognitive pivots to arise, even without explicit curation on our part. The framework further improves performance on PuzzleTrivial by disrupting repetitive reasoning patterns born from strong intuitive priors. Finally, its generalization to GPQA-Diamond remains comparable to vanilla baselines, where failures are bounded by parametric knowledge rather than reasoning flexibility and the mechanism STARS targets is therefore less present. Across these settings, STARS is at or above vanilla in 5 of 6 model-benchmark cells on MATH500 and GPQA-Diamond, with the one exception smaller than any regression shown by either baseline, while DEER and ConCISE fall below vanilla in half of these cells.

Table 2: **Inference efficiency averaged across benchmarks and models.** *Output Token* counts the final sequence, while *Compute Token* counts all generated tokens during decoding.

Method	Output Token	Compute Token	Latency (s)	Throughput (tok/s)
Vanilla	10,940	10,940	286.06	43.86
DEER	10,352	13,988	416.92	35.72
ConCISE	**7,276**	10,722	278.58	34.34
STARS	10,025	**9,981**	**251.34**	**47.09**

Inference efficiency While early-exit baselines appear efficient by output token count, their true computational cost is substantially higher: as shown in Table 2, DEER and ConCISE generate considerably more tokens internally than their final outputs suggest. STARS, which performs

no internal regeneration, achieves the lowest compute token count alongside the best latency and throughput. Detailed analysis is provided in Appendix J.

4.3 Causal analysis: How STARS rectifies reasoning

Table 3: **STARS fix rates by baseline failure mode** on benchmarks where cognitive inertia is the primary failure mode. Plain denotes failures where the correct answer is never reached. p-values from Fisher exact test.

Dataset	Inertia	Plain	Ratio	p
ConditionedMath	33.9%	23.3%	1.46×	0.014
MathPerturb	48.9%	24.6%	1.99×	0.002
PuzzleTrivial	40.0%	24.4%	1.64×	0.008

Table 4: **Prefix continuation analysis** on ConditionedMath. Mean accuracy of the base model over 3 continuation attempts from STARS intervention points before suffix injection, and the proportion never solved in any attempt.

Model	Mean Acc.	Never Solved
DeepSeek-1.5B	63.0%	15.0%
DeepSeek-7B	70.7%	16.3%
Qwen3-4B	84.3%	9.0%

Targeting cognitive inertia An accuracy improvement alone cannot distinguish genuine recovery from an exchange of errors, so we first compare STARS against vanilla per instance across five independent trials: recoveries of vanilla failures outpace regressions of vanilla successes by 1.5–1.7× on ConditionedMath and 1.4–1.8× on PuzzleTrivial. We then ask whether this net gain is tied to our target phenomenon by classifying baseline failures into **Inertia Failures**—where the correct answer appears in the reasoning trace but is subsequently abandoned—and **Plain Failures**. As shown in Table 3, STARS recovers Inertia Failures at significantly higher rates (1.46–1.99×) than Plain Failures across all benchmarks. This confirms that STARS functions as an inertia-specific correction mechanism rather than a general-purpose accuracy booster; see Appendix H.1 for the full analysis.

Causal efficacy of steering Finally, we test whether recovery depends on the steering suffix beyond reaching the intervention point. We perform a *prefix continuation analysis* on ConditionedMath: for each STARS success case, we truncate the trajectory at the first intervention point and allow the model to continue without suffix injection across three independent attempts. As shown in Table 4, even with multiple opportunities, 9–16% of problems remain unsolved without the steering suffix. This confirms that the intervention point alone cannot explain all recoveries, providing causal evidence that the steering suffix contributes to reasoning recovery.

Table 5: **Ablation study on intervention mechanisms** on ConditionedMath. Each row isolates the contribution of individual diagnostic components. See Table 12 for results on AIME25.

	Flip	Recur	Exit	DeepSeek-1.5B		DeepSeek-7B		Qwen3-4B	
				Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
STARS	✓	✓		**53.08**±2.61	9017	**54.62**±1.96	8473	**58.85**±3.12	12923
	✓			<u>51.54</u>±3.92	9499	<u>53.46</u>±4.62	8526	55.77±3.65	12740
		✓		47.31±4.14	8831	48.08±2.11	8763	<u>56.92</u>±1.54	12084
			✓	43.85±2.24	7250	33.08±2.83	3457	56.54±2.88	9366
Vanilla				46.54±4.28	11371	48.08±5.01	9825	56.54±1.54	12629

4.4 Component analysis and ablations

We ablate STARS to investigate the necessity of the individual diagnostic components and spike-aligned intervention timing. This analysis validates the necessity of our three-stage cycle in overcoming the structural persistence of cognitive inertia.

Diagnostic component ablation A natural alternative to active steering is to simply halt generation at detected spikes, which we evaluate as a Spike-based *Early Exit* baseline triggered when next-token confidence exceeds a threshold. As shown in Tables 5 and 12, *Early Exit* substantially reduces generation length but leads to severe accuracy degradation across all models (e.g., a 15% drop on DeepSeek-7B), as spike moments represent states of elevated plasticity where reasoning has not yet

converged—terminating at such points discards potentially recoverable paths. We further evaluate two component variants, *Flip-only* and *Recur-only*: the former yields consistent but incomplete gains, while the latter provides no reliable improvement, confirming that directional correction and loop-breaking address complementary failure modes. Further details are provided in Appendix H.2.

Table 6: **Impact of intervention timing.** Accuracy (%) across ConditionedMath and AIME25. Results show that performance gains depend strongly on spike-aligned timing rather than the presence of a suffix alone.

	ConditionedMath			AIME25		
Method	**DeepSeek-1.5B**	**DeepSeek-7B**	**Qwen3-4B**	**DeepSeek-1.5B**	**DeepSeek-7B**	**Qwen3-4B**
Vanilla	46.54±4.28	48.08±5.01	56.54±1.54	22.67±3.27	39.33±1.33	70.67±5.33
Token-Triggered	44.62±2.24	49.62±3.53	53.85±2.11	21.33±4.52	36.00±1.33	71.33±2.67
Fixed Injection	46.15±2.98	49.23±1.54	53.85±2.11	20.67±2.49	35.33±2.67	71.33±2.67
Verifier (N=512)	45.38±4.96	48.85±5.91	59.62±2.72	**27.33**±4.42	40.00±5.16	72.67±3.89
Verifier (N=1024)	44.62±1.44	50.77±3.35	**60.77**±3.96	20.00±5.58	38.67±4.99	73.33±4.71
Flip-only	51.54±3.92	53.46±4.62	55.77±3.65	23.33±2.11	38.67±5.81	71.33±3.40
STARS	**53.08**±2.61	**54.62**±1.96	58.85±3.12	**27.33**±3.89	**43.33**±2.98	**77.33**±3.89

Importance of intervention timing To test whether performance gains depend on spike-aligned timing rather than the suffix alone, we compare STARS against baselines that apply the identical suffix but at different, non-latent triggers. Token-Triggered injects at specific surface tokens, while Fixed Injection injects at fixed intervals throughout generation; as shown in Table 6, both yield inconsistent gains and often degrade performance relative to vanilla models, indicating that the same suffix does not reliably improve performance when applied at misaligned times. To test a stronger trigger, we additionally implement a surface-text verifier: an LLM-as-a-judge that inspects the generated text every N tokens and intervenes upon a verdict that the reasoning has taken a wrong turn, firing far more often than STARS does and at real API cost. Even this verifier falls below STARS on most model–benchmark pairs and yields inconsistent improvements over vanilla across intervals, further supporting the importance of spike-aligned intervention timing. Further details are provided in Appendices D.3 and H.2.

5 Related works

Prior interpretability work establishes that LLM hidden states encode structured semantics (Azaria & Mitchell, 2023; Burns et al., 2023) and reasoning transitions (Zhao et al., 2025; Chen et al., 2025a), where intervention timing is particularly critical during high-plasticity moments (Zur et al., 2025). Building on the geometry of these dynamics (Zhou et al., 2026; Duan et al., 2026), we identify L_2 displacement spikes as real-time markers of Cognitive Pivots, enabling transition detection without additional training, and treat them as neutral signals to be diagnosed rather than as error indicators in themselves. While various inference-time interventions have emerged to mitigate reasoning rigidity and overthinking (Chen et al., 2025b; Jang et al., 2025), existing approaches often require additional supervision (Li et al., 2023; Zhang et al., 2025) or rely on suppressive heuristics that truncate generation (Yang et al., 2025b; Qiao et al., 2025). In contrast, STARS leverages unsupervised latent spikes as precise anchors to actively redirect erroneous trajectories via state-aware language cues, correcting reasoning paths in real time. Additional discussion of related work is provided in Appendix C.

6 Conclusion

We introduced STARS, a training-free framework that mitigates cognitive inertia by detecting Cognitive Pivots marked by hidden-state L_2 spikes and steering reasoning via a three-stage inference-time pipeline. Experiments across diverse benchmarks show that STARS achieves a superior accuracy-efficiency trade-off compared to existing token-level methods. Two findings generalize beyond the specific framework: a latent spike is a neutral signal rather than an error indicator, so detection must be paired with a diagnostic stage; and *when* to intervene matters more than *what* to inject.

AI USE STATEMENT

We use GPT-5-mini as an automated judge in two analyses: the fixed-interval verifier baseline (Appendix D.3) and the large-scale annotation of detected spikes (Appendix F.1). Beyond these, we used generative AI tools only for writing assistance such as polishing grammar and wording. We have reviewed all AI-assisted text and results for accuracy, and take full responsibility for the final content of this work.

ETHICS STATEMENT

This research conforms with the ICLR Code of Ethics. The work involves no human subjects, crowdsourcing, or sensitive data, and introduces no new datasets or models. We discuss the broader societal impacts of this work, including potential risks of misuse of LLM reasoning technologies in general, in Appendix A.

REPRODUCIBILITY STATEMENT

We provide full experimental details needed to reproduce our main results, including dataset specifications and filtering criteria (Appendix D.1), baseline configurations (Appendix D.2), implementation and compute details (Appendix D.4), and the calibration protocol used to select detection and diagnostic thresholds (Appendix E). All datasets and models used are publicly available and properly cited. Code will be made available upon acceptance.

REFERENCES

Amos Azaria and Tom Mitchell. The internal state of an LLM knows when it's lying. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 967–976, 2023.

Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. Discovering latent knowledge in language models without supervision. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=ETKGuby0hcs.

Runjin Chen, Zhenyu Zhang, Junyuan Hong, Souvik Kundu, and Zhangyang Wang. SEAL: Steerable reasoning calibration of large language models for free. In *Second Conference on Language Modeling*, 2025a. URL https://openreview.net/forum?id=klPszYDIRT.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. Do NOT think that much for 2+3=? on the overthinking of long reasoning models. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025b. URL https://openreview.net/forum?id=MSbU3L7V00.

Zenghao Duan, Liang Pang, Zihao Wei, Wenbin Duan, Yuxin Tian, Shicheng Xu, Jingcheng Deng, Zhiyi Yin, and Xueqi Cheng. Circular reasoning: Understanding self-reinforcing loops in large reasoning models. *arXiv preprint arXiv:2601.05693*, 2026.

Soumya Suvra Ghosal, Souradip Chakraborty, Avinash Reddy, Yifu Lu, Mengdi Wang, Dinesh Manocha, Furong Huang, Mohammad Ghavamzadeh, and Amrit Singh Bedi. Does thinking more always help? mirage of test-time scaling in reasoning models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=tKPqbamNb9.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. Token-budget-aware LLM reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 24842–24855, 2025.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *NeurIPS*, 2021.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *The Twelfth International Conference on Learning Representations*, 2024.

Kaixuan Huang, Jiacheng Guo, Zihao Li, Xiang Ji, Jiawei Ge, Wenzhe Li, Yingqing Guo, Tianle Cai, Hui Yuan, Runzhe Wang, Yue Wu, Ming Yin, Shange Tang, Yangsibo Huang, Chi Jin, Xinyun Chen, Chiyuan Zhang, and Mengdi Wang. MATH-perturb: Benchmarking LLMs' math reasoning abilities against hard perturbations. In *Forty-second International Conference on Machine Learning*, 2025a. URL https://openreview.net/forum?id=OZy70UggXr.

Yao Huang, Huanran Chen, Shouwei Ruan, Yichi Zhang, Xingxing Wei, and Yinpeng Dong. Mitigating overthinking in large reasoning models via manifold steering. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=49Rc51iCso.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. OpenAI o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Doohyuk Jang, Yoonjeon Kim, Chanjae Park, Hyun Ryu, and Eunho Yang. Reasoning model is stubborn: Diagnosing instruction overriding in reasoning models. *arXiv preprint arXiv:2505.17225*, 2025.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36:41451–41530, 2023.

MAA Committees. AIME problems and solutions. https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions.

William J McGuire. Cognitive consistency and attitude change. *The Journal of Abnormal and Social Psychology*, 60(3):345, 1960.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 20275–20321, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1025. URL https://aclanthology.org/2025.emnlp-main.1025/.

ShengYun Peng, Eric Smith, Ivan Evtimov, Song Jiang, Pin-Yu Chen, Hongyuan Zhan, Haozhu Wang, Duen Horng Chau, Mahesh Pasupuleti, and Jianfeng Chi. Large reasoning models learn better alignment from flawed thinking. *arXiv preprint arXiv:2510.00938*, 2025.

Ziqing Qiao, Yongheng Deng, Jiali Zeng, Dong Wang, Lai Wei, Guanbo Wang, Fandong Meng, Jie Zhou, Ju Ren, and Yaoxue Zhang. ConCISE: Confidence-guided compression in step-by-step efficient reasoning. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 8010–8029, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.405. URL https://aclanthology.org/2025.emnlp-main.405/.

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.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. OpenAI GPT-5 system card. *arXiv preprint arXiv:2601.03267*, 2026.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Ulisse Mini, and Monte MacDiarmid. Activation addition: Steering language models without optimization. *arXiv preprint arXiv:2308.10248*, 2023.

Chenlong Wang, Yuanning Feng, Dongping Chen, Zhaoyang Chu, Ranjay Krishna, and Tianyi Zhou. Wait, we don't need to "wait"! removing thinking tokens improves reasoning efficiency. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 7459–7482, 2025. URL https://aclanthology.org/2025.findings-emnlp.394.

Zhaofeng Wu, Linlu Qiu, Alexis Ross, Ekin Akyürek, Boyuan Chen, Bailin Wang, Najoung Kim, Jacob Andreas, and Yoon Kim. Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 1819–1862, 2024. URL https://aclanthology.org/2024.naacl-long.102/.

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*, 2025a.

Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Minghui Chen, Zheng Lin, and Weiping Wang. Dynamic early exit in reasoning models. *arXiv preprint arXiv:2504.15895*, 2025b.

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. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=O6I0Av7683.

Zekai Zhao, Qi Liu, Kun Zhou, Zihan Liu, Yifei Shao, Zhiting Hu, and Biwei Huang. Activation control for efficiently eliciting long chain-of-thought ability of language models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=XNo4yS9n1k.

Yufa Zhou, Yixiao Wang, Xunjian Yin, Shuyan Zhou, and Anru Zhang. The geometry of reasoning: Flowing logics in representation space. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=ixr5Pcabq7.

Amir Zur, Atticus Geiger, Ekdeep Singh Lubana, and Eric Bigelow. Are language models aware of the road not taken? token-level uncertainty and hidden state dynamics. *arXiv preprint arXiv:2511.04527*, 2025.

A BROADER IMPACTS

STARS is a training-free inference-time framework designed to improve the reasoning accuracy and efficiency of large reasoning models. The primary societal benefit of this work lies in reducing unnecessary computation during inference, which may contribute to more efficient and reliable deployment of LLM-based systems. The potential negative societal impacts of this work align with those generally associated with LLM reasoning technologies, including risks of misuse in automated decision-making or generation of misleading content. We emphasize the importance of responsible deployment and adherence to ethical guidelines when applying such methods in practice.

B LIMITATIONS

Despite consistent gains across diverse benchmarks, STARS is subject to two primary failure modes. First, intervention cannot be triggered when the model terminates its reasoning before any salient latent spike occurs, leaving no high-plasticity window available for suffix injection. Second, a detected spike may occasionally steer the model toward an incorrect trajectory rather than correcting it, particularly when the model's parametric priors are exceptionally dominant or the initial reasoning path is fundamentally flawed beyond simple correction. Detailed qualitative examples of both failure modes are provided in Appendix K.

Beyond these failure modes, STARS carries a scope constraint that follows directly from its design: because the detection signal is computed from the decoder's hidden states, the framework requires access to internal representations and is therefore applicable to open-weight models rather than API-only ones. This constraint is shared by latent-monitoring methods generally, and it delineates the deployment setting in which spike-triggered steering can currently be applied.

C RELATED WORKS

Large reasoning models The paradigm of LLMs has evolved from scaling parameters to scaling test-time compute. Recent research has shown that allocating more inference budget—specifically by extending reasoning chains—can yield performance gains comparable to massive parameter scaling (e.g., s1 (Muennighoff et al., 2025)). State-of-the-art models like o1 (Jaech et al., 2024) and DeepSeek-R1 (Guo et al., 2025) operationalize this principle, employing extended "Chain of Thought" (CoT) strategies to decompose intricate problems and perform self-correction. However, this test-time scaling introduces new challenges: as reasoning chains lengthen without guidance, models become prone to inefficiencies where the computational cost does not strictly correlate with performance gains, necessitating mechanisms to control and optimize the reasoning process.

Hidden state dynamics and reasoning transition Prior interpretability work has established that LLMs encode structured semantic information within their latent space, including truthfulness and factual validity, often decoupled from surface-level generated text (Azaria & Mitchell, 2023; Burns et al., 2023). More recently, Zhao et al. (2025) and Chen et al. (2025a) have shown that hidden state dynamics carry structured signals specifically relevant to reasoning transitions, establishing that the internal trajectory during chain-of-thought generation reflects the functional structure of the reasoning process itself. Building on this foundation, Zur et al. (2025) further find that the timing of intervention is critical—guidance is most consequential precisely when the model has not yet committed to a particular reasoning direction. A complementary line of work characterizes the geometry of these dynamics directly. Zhou et al. (2026) formalize reasoning as a flow through representation space and establish that logical structure governs both the magnitude and the direction of its velocity, which grounds the two quantities STARS monitors: the L_2 displacement measures that velocity, and the cosine between consecutive update vectors measures the direction component whose reversal we treat as a conflict signal. Duan et al. (2026) analyze self-reinforcing loops and find that representations across repetition cycles contract toward near-identical geometry, indicating that looping is detectable in the latent state before it surfaces as repeated text—the property our recurrence score exploits. We build on these insights by identifying L_2 displacement spikes as a concrete, real-time signal of such transition moments, enabling Cognitive Pivots to be detected without additional training. Importantly, we use these quantities not as error indicators but as tentative signals to be diagnosed: because spikes

are semantically neutral (Section 2.3), STARS locates candidates by magnitude and then resolves them by directional and recurrence geometry, rather than thresholding any single signal.

Inference-time intervention As reasoning chains lengthen, models become increasingly susceptible to parametric priors that override task-specific constraints—a failure that self-correction alone often cannot resolve (Huang et al., 2024; Wu et al., 2024). Recent work has characterized these failures as overthinking and reasoning rigidity (Chen et al., 2025b; Ghosal et al., 2025; Jang et al., 2025), motivating inference-time methods that actively regulate the reasoning trajectory. Existing approaches include activation-level steering that modifies internal representations to suppress overthinking (Li et al., 2023; Turner et al., 2023; Huang et al., 2025b), hidden-state probing methods that predict answer correctness to guide early exit (Zhang et al., 2025), and efficiency-driven methods that truncate generation via token-level heuristics (Yang et al., 2025b; Qiao et al., 2025). While these methods share the goal of regulating reasoning, they either require additional supervision or remain fundamentally suppressive—capable of halting generation but unable to actively redirect an incorrect trajectory. STARS instead leverages latent L_2 spikes as precise intervention anchors and steers via lightweight natural-language cues, correcting erroneous trajectories in real time without additional training.

D Extended experimental setup

D.1 Dataset specifications & filtering

We evaluate STARS on a collection of reasoning benchmarks covering complementary failure modes of large reasoning models.

ConditionedMath and PuzzleTrivial are derived from the ReasoningTrap benchmark (Jang et al., 2025), which is designed to probe reasoning rigidity under counter-intuitive or disrupted constraints. Within this framework, ConditionedMath modifies MATH- and AIME-style mathematical problems by introducing explicit counterfactual conditions, while PuzzleTrivial adapts well-known logic puzzles to invalidate intuitive default solution strategies. For both datasets, we filter out instances with ambiguous or ill-defined ground truth, as well as near-duplicate problem structures, to ensure reliable evaluation. After filtering, we retain 52 instances for ConditionedMath and 33 representative problems for PuzzleTrivial.

MathPerturb (Huang et al., 2025a) is derived from Level-5 problems in MATH500 and targets perturbation-induced instability in mathematical reasoning. We remove training-set overlaps and instances with undefined or non-unique answers, resulting in 77 evaluation samples.

All samples excluded during the filtering process for ConditionedMath and MathPerturb are not discarded, but are instead reserved exclusively for calibration.

MATH500 (Hendrycks et al., 2021) is a standard benchmark for mathematical problem solving, covering a broad range of difficulty levels and problem types. We use the full 500-problem evaluation set without additional filtering, as it serves as a general-purpose reference benchmark rather than an inertia-specific diagnostic.

To assess generalization under extended reasoning horizons, we additionally evaluate on AIME24 and AIME25 (MAA Committees), which consist of held-out competition-style problems and are strictly disjoint from all calibration data. Beyond mathematical reasoning, we further evaluate domain generalization on GPQA-Diamond (Rein et al., 2024), a high-difficulty subset of GPQA focusing on graduate-level scientific reasoning.

D.2 Detailed baseline configurations

We compare STARS against representative **training-free** inference-time baselines under identical decoding settings. All methods operate without any additional fine-tuning, enabling controlled comparison on the same model checkpoints.

DEER (Yang et al., 2025b) is a training-free inference-time framework designed to reduce redundant reasoning by dynamically deciding when to terminate generation. During decoding, DEER periodically induces a *trial answer* from the current reasoning state and evaluates its reliability

using token-level uncertainty signals. If the induced answer exhibits sufficiently high confidence, generation is terminated early and the answer is returned; otherwise, decoding continues. To mitigate the additional overhead introduced by trial answer induction and confidence evaluation, DEER further proposes a *branch-parallel decoding* strategy. This approach linearizes multiple candidate branches into a single sequence using specialized causal attention masks, allowing multiple trial answers to be evaluated within a single forward pass. While effective at amortizing the cost of confidence probing, this mechanism requires maintaining multiple concurrent branches and incurs increased memory usage and implementation complexity.

In our experiments, we evaluate the standard single-trajectory configuration of DEER without branch-parallel acceleration. This setting reflects a common deployment scenario where GPU memory constraints or system simplicity limit the feasibility of maintaining parallel reasoning branches. Specifically, we apply entropy-based early stopping with a threshold of 0.95, following the default inference-time configuration of the method.

ConCISE (Qiao et al., 2025) performs confidence-guided step-wise compression by actively steering the generation process. During decoding, ConCISE detects reflection steps and injects confidence phrases to regenerate the current step, effectively preventing low-utility reflective reasoning from being incorporated into the final chain. In addition, it applies confidence-based early stopping once a sufficiently confident answer is produced, resulting in a compact final reasoning trace while preserving the standard answer format.

Importantly, while the final output produced by ConCISE is substantially shorter, this compression is achieved through regeneration rather than simple truncation. Reflection steps are invalidated and re-generated with confidence injection, meaning that the total number of tokens generated internally during decoding can exceed the length of the final reasoning trace.

Although ConCISE is originally proposed as a training framework that combines confidence injection with early stopping, we adopt its *training-free decoding variant* (ConCISE-Decoding) for a controlled inference-time comparison. In this setting, confidence estimation and regeneration are applied directly during decoding, without relying on ground-truth verification or additional supervision. We use step-wise confidence-guided decoding with an early-stopping threshold of 0.5, following the commonly evaluated threshold regime reported in the original work.

D.3 Fixed-interval verifier

The verifier baseline is designed to isolate the *trigger* while holding the *intervention* fixed. STARS and this verifier target the same event—the moment at which reasoning takes a wrong turn—but read it from different sources: STARS from the latent trajectory (an L_2 spike whose update direction reverses), the verifier from the surface text.

We use GPT-5-mini (Singh et al., 2026) as an untrained judge that inspects the freshly generated text at fixed intervals of N tokens and returns a binary verdict on whether the reasoning has taken a wrong turn. On an affirmative verdict, the identical *shifting suffix* used by STARS is appended, after which decoding continues normally; all other decoding settings match those of the other methods. We evaluate $N = 512$, roughly the smallest span a judge can meaningfully assess in context, and $N = 1024$ as a more conservative interval. Both settings inspect the trajectory substantially more often than STARS intervenes (a mean of 3.35 flips per sequence across models), so the comparison is deliberately generous to the verifier. Even so, the verifier calls an external judge repeatedly throughout decoding: even with a deliberately inexpensive judge it incurs 1.3–1.5 USD of API cost per run and an average end-to-end latency of 518.01s, more than twice the 251.34s of STARS (Table 2). By contrast, the STARS trigger is the L_2 norm of a hidden state already computed during the ordinary forward pass, requiring no additional forward pass, no external model, and no API cost. The resulting accuracy and cost are reported in Table 6 and discussed in Section 4.4.

D.4 Implementation details

All experiments are conducted using the HuggingFace transformers library. We use the default decoding parameters (temperature and top-p) provided by each model to preserve their intended reasoning behavior. The maximum sequence length is set to 32k tokens for all experiments. Each reported result is averaged over 5 independent runs with different random seeds, and we report the

Default Prompt

```
Please reason step by step, and put your final answer within \boxed{}.
```

Figure 5: The default prompt shared across all methods and benchmarks.

mean accuracy along with the standard deviation. All methods share the same default prompt shown in Figure 5, unless otherwise specified. All experiments are performed on a single NVIDIA RTX 6000 Ada GPU (48GB VRAM).

Hidden state convention For the monitored signal we use the post-block residual stream at layer l, i.e., the tensor exposed by HuggingFace as `hidden_states[layer+1]`, taken before any subsequent normalization is applied. All displacement, cosine, and recurrence quantities in Sections 3.1 and 3.2 are computed on this tensor, so the detection signal reflects the residual stream as it is carried forward rather than a normalized view of it.

Suffix injection mechanism Decoding follows a standard autoregressive loop with KV caching, and STARS leaves that loop intact. When an intervention is triggered, the suffix is tokenized and appended as ordinary tokens through a single forward pass that *extends* the KV cache rather than editing or invalidating it; generation then resumes from the extended context. No hidden state is modified, no previously generated token is discarded, and no branch is maintained, which is why suffix injection adds only the cost of the suffix tokens themselves and preserves the standard decoding flow (Appendix J).

E Framework calibration and sensitivity

This section details the calibration protocol used to establish the signal-driven detection parameters $(l^*, \tau^{(l^*)})$. All calibration steps are conducted exclusively on held-out samples from ConditionedMath and MathPerturb, which are strictly disjoint from the test sets used for reporting evaluation results (Section D.1). This separation ensures that calibration does not introduce information leakage into the final evaluation.

E.1 Calibration protocol

To ensure reliable spike detection across heterogeneous model architectures, we adopt a data-driven calibration pipeline that identifies the most stable and informative signal source, rather than fixing layers a priori.

Selection procedure For each model, we perform a parameter sweep over a range of sensitivity multipliers $k \in \{5, 7, \ldots, 13\}$ using a calibration dataset. The optimal configuration is selected based on a three-step hierarchy:

1. **Density Filter:** We first discard any configuration that fails to meet the minimum density constraint (less than 1 event per 1k tokens), ensuring that selected signals reflect recurring reasoning dynamics rather than incidental numerical fluctuations.
2. **Signal Maximization:** Among the valid candidates, we rank configurations by their **Spike Prominence Ratio (SPR)**.
3. **Stability Check (Majority Voting):** We assess the stability of SPR rankings across independent trials via majority voting. This criterion is used as a secondary indicator of robustness rather than a hard selection constraint.

E.2 Selected configurations

Table 7 presents the final calibrated parameters. The results reveal distinct structural preferences based on model capacity.

Table 7: **Calibrated detection parameters.** Comparing the primary (top row) and secondary (bottom row) configurations illustrates the stability of layer-wise signal selection across calibration trials.

Model	Layer	Thr (τ)	SPR	Vote
DeepSeek-1.5B	L27	1.730	9.29	0.40
	L21	1.460	8.80	0.60
DeepSeek-7B	L13	1.509	160.6	0.80
	L13	1.433	157.6	0.60
Qwen3-4B	L2	2.245	54.66	0.89
	L2	2.023	54.41	1.00

For high-capacity models like **DeepSeek-7B** and **Qwen3-4B**, the selection converges robustly to a single layer with exceptionally distinct signals, indicating a centralized "reasoning hub". In sharp contrast, the smaller **DeepSeek-1.5B** exhibits significantly weaker spike signals (SPR < 10) and a lack of convergence in layer selection (the vote is split between Layer 27 and Layer 21). This suggests that in smaller models, the reasoning process is more distributed and less distinguishable from background generation noise.

Based on this analysis, we adopt the SPR-optimal configuration among density-valid candidates (corresponding to the first row for each model) as the fixed detection standard for all subsequent experiments. When SPR ranking and vote stability disagree (e.g., DeepSeek-1.5B), we prioritize SPR while reporting the runner-up configuration for transparency.

E.3 Suffix design

The suffixes used in STARS are designed as lightweight, state-aware cues rather than corrective instructions. They do not inject external knowledge or prescribe a specific solution path, but instead reflect the diagnosed reasoning state back to the model.

Adaptive Steering Suffixes

Shifting Suffix:

"Wait. I am shifting my reasoning. Let's double-check if this direction is valid and grounded in the text."

Loop Breaker Suffix:

"Wait. This line of thinking is looping. Let's pause and pivot. Is there a simpler way to look at this problem?"

Figure 6: The state-aware suffixes injected by STARS.

Figure 6 illustrates the two suffix types used in STARS. The *Shifting Suffix* is injected when a sudden directional change is detected, prompting the model to reassess whether the new trajectory is grounded in the problem context. In contrast, the *Loop Breaker Suffix* is triggered upon detecting a recurrent reasoning state, explicitly surfacing repetition and encouraging exploration of alternative trajectories. Although both suffixes are expressed as short linguistic cues, they are differentiated by the underlying reasoning state they acknowledge.

Table 8: **Suffix structure ablation** on ConditionedMath using DeepSeek-7B.

Suffix Variant	State	Action	Acc.
STARS (Full)	✓	✓	**54.62**$\pm$1.96
State-only	✓	–	50.38$\pm$2.83
Action-only	–	✓	51.15$\pm$2.61
Suffix-swap	✓	✓	<u>53.85</u>$\pm$3.22

We further investigate whether the efficacy of these cues depends on their functional semantics rather than specific lexical forms by ablating the suffix into two components: *state acknowledgment* and

action guidance. As detailed in Table 8, removing either component degrades performance, demonstrating that neither element alone provides sufficient steering to disrupt the model's inertial trajectory. The *suffix-swap* variant—injecting mismatched cues for a diagnosed state—attains intermediate performance, suggesting that while structural components are inherently beneficial at high-plasticity spike points, optimal steering requires **coherent semantic alignment** between the diagnosed state and its corresponding guidance. Peak accuracy is only achieved when the model's internal state and the external cue are perfectly synchronized.

Table 9: Directional flip statistics under varying τ_{flip}. p_{flip} denotes the proportion of spikes classified as directional flips ($c_t < -\tau_{\text{flip}}$).

τ_{flip}	p_{flip}	d'
0.05	0.65	1.97
0.10	0.49	2.00
0.15	0.34	2.08
0.20	0.22	2.20
0.25	0.13	2.48
0.30	0.08	3.12
0.35	0.06	3.71
0.40	0.05	4.26

E.4 Threshold selection and robustness

Flip threshold To support our choice of the flip threshold τ_{flip}, we analyze its behavior on the same held-out calibration set described in Appendix E, avoiding data snooping on the evaluation benchmarks. Rather than explicitly optimizing this parameter for performance, we analyze the distributional characteristics of flip signals and select a conservative operating point.

We quantify the separation between flipped and non-flipped updates using the standardized effect size d':

$$d' = \frac{\mu_{\text{non-flip}} - \mu_{\text{flip}}}{\sqrt{\frac{1}{2}(\sigma^2_{\text{non-flip}} + \sigma^2_{\text{flip}})}} \tag{8}$$

where μ and σ denote the mean and standard deviation of cosine similarities within each group.

Table 9 reports summary statistics of the flip signal collected at spike locations. The table illustrates how the proportion of detected flips varies as a function of τ_{flip}.

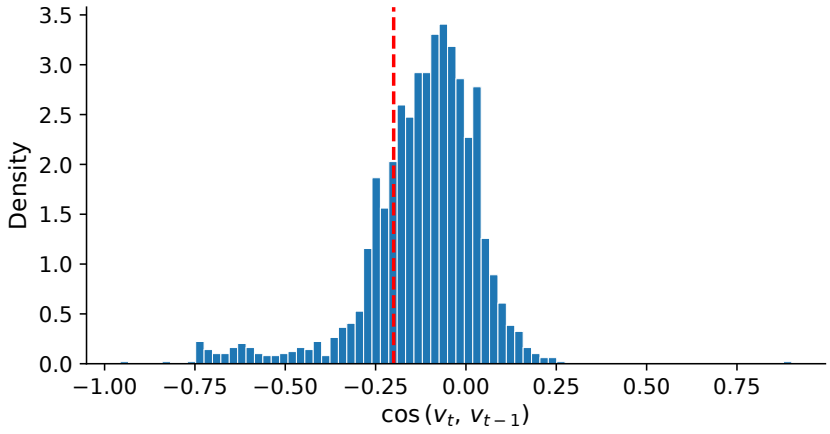


Figure 7: Histogram of flip scores ($\cos(v_t, v_{t-1})$) at detected spike locations. The dashed red line indicates the selected flip threshold $-\tau_{\text{flip}}$ (here, $\tau_{\text{flip}} = 0.2$), used to identify strong directional reversals.

Figure 7 visualizes the empirical distribution of flip scores. The distribution is sharply concentrated around zero, with a substantial mass near small negative values. This suggests that most spike-associated updates correspond to weak directional changes, while only a subset exhibits strong reversals beyond the operational flip threshold.

Based on this distribution, we set $\tau_{\text{flip}} = 0.2$. As shown in Table 9, this value corresponds to a regime where the separation between flipped and non-flipped updates is already substantial ($d' \approx 2.2$), while the proportion of detected flips remains moderate. In particular, $\tau_{\text{flip}} = 0.2$ classifies roughly one-fifth of spike events as directional flips, filtering out minor oscillations near zero cosine similarity without restricting detection to only extreme reversals. This choice therefore represents a conservative operating point that balances discriminability and intervention frequency.

Recurrence threshold The recurrence threshold is likewise grounded in the distribution of the underlying signal rather than selected to maximize a downstream metric. On DeepSeek-1.5B, the recurrence score—the cosine similarity between the hidden state at a pivot and previously visited pivot states—exhibits a bimodal pattern: a low-similarity cluster corresponding to genuinely new pivots and a high-similarity cluster corresponding to recurrent states, with the density thinning around 0.9. We set $\tau_{\text{recur}} = 0.9$ at this low-density region, which separates newly visited states from returns to highly similar states; a substantially lower value would begin to label loosely related states as recurrences. The value is deliberately conservative in the sense that cosine similarity approaches 1 only as two states become nearly collinear, so 0.9 admits only highly similar latent states as evidence of a loop. We fix this value based on DeepSeek-1.5B and apply it to the remaining models without re-tuning, where it remains near the low-density region separating the two clusters. The sensitivity analysis in Table 10 additionally shows that performance does not hinge sharply on this particular value.

Joint threshold robustness We next analyze the robustness of STARS with respect to its steering thresholds, namely the flip threshold τ_{flip} and the recurrence threshold τ_{recur}. Table 10 reports accuracy over a small grid of threshold combinations.

Table 10: Robustness analysis over Flip and Recurrence thresholds using DeepSeek-7B.

τ_{flip}	τ_{recur}	**CondMath**	**AIME24**
0.1	0.5	51.54±4.93	52.67±4.90
	0.7	51.54±2.83	52.67±4.90
	0.9	50.38±2.24	52.00±5.81
0.2	0.5	54.48±4.53	50.67±5.73
	0.7	53.08±3.12	51.33±4.00
	0.9	**54.62**±1.96	**55.33**±3.40
0.3	0.5	54.32±1.59	52.00±2.67
	0.7	52.69±3.96	54.00±1.33
	0.9	50.77±5.91	54.00±3.89

As shown in the table, performance varies non-trivially across threshold settings and different datasets favor different operating regimes. Lower thresholds tend to trigger frequent interventions, which can improve responsiveness to cognitive inertia but risk destabilizing otherwise correct reasoning. Conversely, overly conservative thresholds reduce unnecessary interventions but may fail to correct persistent reasoning loops.

We therefore select the default configuration ($\tau_{\text{flip}} = 0.2$, $\tau_{\text{recur}} = 0.9$) as it achieves a favorable balance across datasets for DeepSeek-7B. This setting consistently mitigates cognitive inertia on ConditionedMath while simultaneously maintaining strong generalization performance on AIME24. Importantly, this choice was not selected to optimize any single benchmark in isolation, but represents the most stable trade-off observed under this model setting across distinct reasoning regimes. Overall, these results indicate that while STARS is sensitive to threshold choices, its effectiveness does not depend on finely tuned parameters. Rather than prescribing a universal optimum, this analysis suggests that robust performance emerges from operating within a reasonable regime that balances inertia correction and generalization.

Cross-domain robustness Notably, these thresholds are calibrated exclusively on held-out mathematical samples yet remain effective across out-of-domain benchmarks including science reasoning (GPQA-Diamond) and logical reasoning (PuzzleTrivial) without any domain-specific adjustment, confirming that the calibrated parameters generalize beyond the mathematical domain.

This property is worth stating explicitly, because it distinguishes a general phenomenon from a detector engineered for a particular evaluation setting. Our protocol is a cross-domain transfer test rather than a per-benchmark fit: every parameter is calibrated once on held-out samples disjoint from all test sets, then frozen and applied unchanged to domains that are never observed during calibration. A detector fitted to the idiosyncrasies of these benchmarks would be expected to degrade precisely under this shift, since the target domains are absent from the calibration data. The same stability holds across models: the flip–failure relationship is significant in all three model families (Appendix G), and the critical-pivot proportion remains consistent between the mathematical and logical domains (Appendix F.1). The transfer of frozen parameters across both domain and model shifts therefore points to a property of LRM reasoning dynamics rather than an artifact of the specific benchmarks or architectures used for calibration.

F Analysis of latent spike dynamics

F.1 Manual and large-scale annotation of latent spikes

To empirically validate the semantic implications of latent spikes proposed in Section 2.3, we conducted a granular qualitative analysis on the **ConditionedMath** benchmark, focusing on the three representative samples exhibiting the highest spike density for each model. Through manual annotation of 151 latent spikes isolated from these high-inertia trajectories, we observed that high-magnitude updates are predominantly *Functional Pivots* (110 instances, 72.8%), representing necessary operational shifts such as calculation initiation or symbolic derivation. In contrast, *Critical Pivots*—markers of genuine inertia risk such as self-doubt or loop onset—constitute the minority (41 instances, 27.2%). To further assess whether this distribution is domain-agnostic, we extended the same annotation methodology to **PuzzleTrivial**, a logical reasoning benchmark from a distinct domain. Applying identical labeling criteria across 176 spike windows from three model families, we observed a Critical Pivot proportion of 32.2%—closely consistent with the 27.2% observed on ConditionedMath. This cross-dataset consistency suggests that the fundamental neutrality of latent spikes is a characteristic of LRM reasoning dynamics rather than an artifact of a single benchmark or domain.

Scaling the annotation with an LLM judge Manual annotation inherently bounds how many spikes can be examined, so we additionally scaled the identical labeling protocol using an LLM judge. We prompt GPT-5-mini in a few-shot setting, supplying the 327 manually annotated spikes above (151 from ConditionedMath and 176 from PuzzleTrivial) as labeled exemplars alongside the same labeling criteria given to human annotators, and apply it to 2,068 detected spikes—approximately a $6\times$ increase over the full manual set and a $14\times$ increase over the initial ConditionedMath analysis. The large-scale result reproduces the manual finding: *Critical Pivots* remain a minority, averaging roughly 35% across models, with benign *Functional Pivots* forming the majority. We emphasize that the role of this analysis is descriptive. It supports the claim in Section 2.3 that latent spikes are semantically neutral—predominantly benign and only partly pathological, and therefore not usable as error indicators on their own—which is exactly why STARS does not act on spike magnitude alone but adopts a two-stage detect-then-diagnose design. The efficacy of the method itself rests on the ablations in Section 4.4 rather than on these annotations.

Implications for spike interpretation Crucially, as visualized in Figure 3, Functional and Critical Pivots do not exhibit a clear separation when examined through text-aligned latent magnitudes alone. Despite their opposing semantic interpretations, both Functional and Critical Pivots often manifest as comparably high-magnitude updates in the hidden space. This misalignment highlights a fundamental limitation of window-based, text-centric analyses: the semantic role of a pivot, as perceived by human annotators, does not necessarily coincide with the precise moment of internal state transition within the model. Consequently, interventions triggered purely by surface-level cues or post-hoc textual symptoms risk being mistimed, either suppressing valid reasoning steps or reacting after the trajectory

has already diverged. Motivated by this observation, STARS treats latent spikes not as errors to be corrected, but as temporal anchors for high-plasticity transitions, enabling subsequent trajectory-level diagnosis that operates directly on hidden-state dynamics rather than textual proxies.

F.2 Spike and flip distribution across model scales

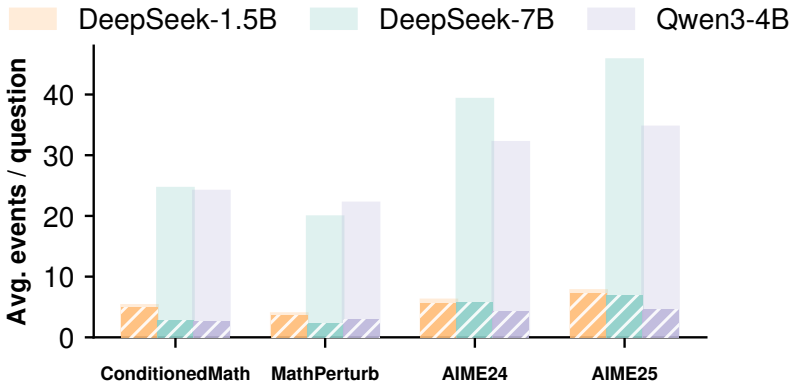


Figure 8: **Average spike and flip counts across model scales.** Semi-transparent bars indicate the average number of latent spikes detected per question, whereas solid bars denote directional flips, a structurally meaningful subset of spike events.

Figure 8 shows how model scale influences the latent dynamics available for spike-triggered intervention across benchmarks. We observe that model scale and architecture jointly influence the frequency of latent spike events, reflecting a richer space of candidate reasoning trajectories and more frequent high-plasticity transitions. Despite this increase in overall spike frequency, only a limited subset of spike events is identified as directional flips. Importantly, the number of such flips remains relatively constrained across model scales, indicating that the proposed flip-based diagnosis selectively isolates structurally significant pivots rather than reacting to spike frequency alone. The same pattern holds for the smaller DeepSeek-1.5B model: although spike activity is sparser due to a more compressed and less dynamically expressive reasoning process, flip-based diagnosis still identifies a small but meaningful subset of structurally significant transitions. Together, these observations suggest that STARS operates as a robust selection mechanism, maintaining selective and stable intervention behavior even as the underlying latent dynamics vary substantially with model scale.

G Directional flip count as an inertia severity indicator

The STARS framework rests on the premise that directional flips—abrupt reversals in the hidden-state update direction—are meaningfully correlated with cognitive inertia and, consequently, with reasoning failure. To empirically validate this premise, we analyze whether the total number of directional flips detected per trajectory is predictive of the final reasoning outcome, without conditioning on whether an intervention was applied. For each trajectory produced by STARS, we record the total flip count and stratify trajectories by final correctness. Table 11 reports mean flip counts for correct and incorrect trajectories on ConditionedMath and MathPerturb combined, stratified by model.

Table 11: **Directional flip counts** (mean $\pm$ standard error) for correct vs. incorrect trajectories, combined across ConditionedMath and MathPerturb. p-values from Mann–Whitney U test.

Model	Correct	Wrong	p-value
DeepSeek-1.5B	3.06 ± 0.39	5.48 ± 0.57	$< 10^{-8}$
DeepSeek-7B	1.80 ± 0.12	4.28 ± 0.44	$< 10^{-13}$
Qwen3-4B	2.60 ± 0.09	3.15 ± 0.21	0.015

Across all three model families, incorrect trajectories consistently accumulate substantially more directional conflicts than correct ones, in all cases reaching statistical significance. The effect is particularly pronounced for DeepSeek-7B, where wrong trajectories exhibit $2.4\times$ more flips than correct ones (4.28 vs. 1.80, $p < 10^{-13}$). This finding supports the cognitive inertia hypothesis at a quantitative level: the frequency of latent directional conflict is a reliable proxy for inertia severity, and trajectories that undergo persistent hidden-state reversals are less likely to recover a stable reasoning path, even under active steering.

The relationship between flip count and correctness varies across benchmarks in a manner consistent with the inertia hypothesis. On GPQA-Diamond, where failures primarily reflect the limits of parametric knowledge rather than reasoning flexibility, the difference in flip counts between correct and incorrect trajectories is negligible (5.66 vs. 5.93 overall), consistent with a knowledge-bounded rather than inertia-bounded failure mode. The cross-benchmark pattern thus confirms that flip count tracks inertia-specific failure rather than general reasoning difficulty, validating the diagnostic rationale of STARS: by targeting moments of high directional conflict, the framework intervenes precisely at the moments most predictive of reasoning failure.

H Supplementary quantitative studies

H.1 Per-instance failure mode analysis

Per-instance results Across 5 independent trials, we count cases where STARS recovers a problem that vanilla decoding fails to solve (**TP gain**) versus cases where vanilla succeeds but STARS fails (**FP loss**). On benchmarks where cognitive inertia is the primary failure mode, TP gains consistently outnumber FP losses—on ConditionedMath by approximately 1.5–$1.7\times$, on PuzzleTrivial by approximately 1.4–$1.8\times$. On GPQA-Diamond, where the primary bottleneck is knowledge limitation rather than reasoning flexibility, the net effect is more modest. While STARS occasionally intervenes during benign Functional Pivots, its flip-and-recurrence diagnosis imposes a stricter intervention criterion than spike detection alone, and prefix-continuation results indicate that the steering suffix contributes to recovery beyond the preceding reasoning trajectory alone.

Distinguishing inertia failures from plain failures A core hypothesis underlying STARS is that cognitive inertia manifests as a failure to commit to a correct reasoning path that the model has already identified, rather than a failure to find the answer altogether. To operationalize this distinction, we classify each incorrect trajectory as either an **Inertia Failure** or a **Plain Failure** based on whether the ground-truth answer appears anywhere in the reasoning trace prior to the final answer. If the correct answer is present but ultimately abandoned in favor of an incorrect conclusion, we treat this as evidence of inertia-driven failure. For cases where the ground-truth answer is a low-magnitude integer (e.g., single-digit values) that may arise naturally as an intermediate computation, we apply additional manual verification to prevent false positive classifications.

Benchmark selection This analysis is restricted to ConditionedMath, MathPerturb, and PuzzleTrivial, as these benchmarks are specifically designed to elicit cognitive inertia by introducing counterfactual conditions or disrupted intuitive priors that directly trigger parametric-prior-driven failures. Knowledge-bounded benchmarks such as GPQA-Diamond are excluded because failures primarily reflect the limits of parametric knowledge rather than reasoning flexibility, making the Inertia Failure criterion ill-suited as a diagnostic signal. AIME and Math500 benchmarks are similarly excluded, as the proportion of inertia-type failures among baseline errors is insufficient to yield a meaningful contrast between failure modes.

Representative examples are provided in Appendix K: Figures 10–13 illustrate Inertia Failure recoveries, while Figures 14 and 15 show a Plain Failure case where STARS also succeeds, albeit less consistently across runs.

H.2 Extended ablation study results

Diagnostic component ablation **Flip-only** injects the shifting suffix at every detected directional flip with recurrence handling disabled, while **Recur-only** injects the loop breaker suffix at recurrence-detected spikes with flip-based steering disabled. **Early Exit** instead terminates generation at a

Table 12: **Ablation study on intervention mechanisms** on AIME25.

	Flip	Recur	Exit	DeepSeek-1.5B		DeepSeek-7B		Qwen3-4B	
				Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
STARS	✓	✓		**27.33**±3.89	14708	**43.33**±2.98	12241	**77.33**±3.89	21069
	✓			23.33±2.11	15387	38.67±5.81	13820	71.33±3.40	21876
		✓		22.67±3.89	14017	38.00±4.52	13198	68.67±5.42	22018
			✓	22.67±3.89	13174	28.67±4.52	9239	68.00±5.81	16953
Vanilla				22.67±3.27	16192	39.33±1.33	14060	70.67±5.33	21064

detected spike once next-token confidence exceeds a threshold, $\mathcal{M}_t = \max_y p(y \mid h_t) > 0.9$. Table 12 presents these variants on AIME25, complementing the ConditionedMath results reported in Table 5, and confirms that the relative contributions of each diagnostic component are consistent across both benchmarks.

Intervention timing baselines **Token-Triggered** injects the suffix at "wait" tokens—a reflection token widely used in prior work on reasoning control (Yang et al., 2025b; Wang et al., 2025)—up to 3 times per sequence, matched to the average flip frequency observed across models (mean 3.35 flips). **Fixed Injection** injects the suffix at fixed token positions (2k, 4k, and 8k tokens), designed to be entirely independent of the model's reasoning content.

I Effect of explicit instruction prompting

As a complementary point of comparison, we examine the effect of *static prompt-based instruction*, which attempts to influence reasoning behavior solely through input-level linguistic cues.

Prompt Hinting Instruction

```
Please reason step by step, and put your final answer within \boxed{}.
An unusual condition is given.  Do not overinterpret.
```

Figure 9: The instruction used for Prompt Hinting.

As illustrated in Figure 9, Prompt Hinting prepends an explicit cautionary instruction designed to discourage overinterpretation, but applies no intervention once decoding begins. This form of input-level guidance has been shown to yield measurable gains on benchmarks targeting *reasoning rigidity*, where failures are often driven by early misinterpretation of counterfactual or unusual constraints (Jang et al., 2025).

A key limitation of this approach lies in its timing. Because the instruction is provided only at the input stage, its influence is restricted to shaping the initial reasoning trajectory. Once generation progresses and the model's internal reasoning dynamics begin to drift under strong parametric priors, static instructions are frequently overridden, with no mechanism for mid-course correction.

Table 13 reports a quantitative comparison between Prompt Hinting, vanilla decoding, and STARS on ConditionedMath, AIME24, and AIME25 across three model scales. Prompt Hinting exhibits mixed behavior across settings. Notably, on ConditionedMath with Qwen3-4B, static instruction yields a substantial accuracy improvement, suggesting that when reasoning failures primarily stem from early misinterpretation, pre-conditioning the model can be effective. In this regime, the initial bias introduced by the instruction is sufficient to guide the model toward a more appropriate reasoning path.

However, this effect does not generalize consistently. On larger models such as DeepSeek-7B, and on longer-horizon benchmarks such as AIME24 and AIME25, Prompt Hinting provides limited or unstable gains, and in some cases underperforms vanilla decoding. These results indicate that static instruction alone is insufficient when reasoning errors arise from downstream instability, repetition, or inertia that emerges during generation.

Table 13: **Effect of prompt hinting.** Comparison between explicit instruction prompting and dynamic steering on ConditionedMath, AIME24, and AIME25.

Method	CondMath		AIME24		AIME25	
	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
DeepSeek-1.5B						
Vanilla	46.54±4.28	11371	**28.00**±4.52	17136	22.67±3.27	16192
Prompt	50.38±4.77	10017	26.67±5.96	16968	23.33±2.98	14947
STARS	**53.08**±2.61	9018	**28.00**±4.00	15014	**27.33**±3.89	14708
DeepSeek-7B						
Vanilla	48.08±5.01	9825	53.33±2.11	13010	39.33±1.33	14060
Prompt	47.69±4.28	9913	49.33±5.73	13298	38.00±1.63	13396
STARS	**54.62**±1.96	8474	**55.33**±3.40	12199	**43.33**±2.98	12241
Qwen3-4B						
Vanilla	56.54±1.54	12630	76.00±3.89	19595	70.67±5.33	21064
Prompt	**67.69**±4.11	11210	75.33±4.52	19362	76.00±2.49	20751
STARS	58.85±3.12	12924	**77.33**±3.89	19120	**77.33**±3.89	21069

On AIME24 in particular, static instruction improves none of the three models and degrades DeepSeek-7B by 4.0 points, while STARS remains at or above vanilla on every model. The advantage of prompt hinting is therefore narrow along two axes. It is *model-dependent*: on ConditionedMath, where prompting is highly effective for Qwen3-4B, it already fails to help DeepSeek-7B. It is also *task-dependent*: the same instruction that helps on ConditionedMath does not consistently improve performance on AIME24 and AIME25. The inconsistency is therefore a property of static prompting itself rather than evidence that the gains from spike-triggered steering are unstable—prompting acts once, before any reasoning has occurred, and cannot respond to instability that emerges later in the trajectory.

Overall, this analysis highlights that explicit instruction can influence reasoning behavior, but its effectiveness is highly sensitive to model scale and task structure, and fundamentally constrained by its lack of temporal adaptivity. This motivates the use of inference-time methods such as STARS, which intervene selectively at moments of detected instability rather than relying solely on input-level guidance.

J Inference efficiency analysis

J.1 Decoding overhead analysis

While output token length is commonly used as a proxy for inference cost, it can substantially underestimate the true computational overhead for methods that involve internal token generation beyond the final trace. Table 14 reports output and compute tokens averaged across all benchmarks per model for DEER and ConCISE, revealing a consistent gap that varies across model architectures.

Table 14: **Decoding overhead by model.** Output tokens denote the final trace length; compute tokens reflect total tokens generated during decoding including internally discarded tokens.

	DeepSeek-1.5B		DeepSeek-7B		Qwen3-4B	
	DEER	ConCISE	DEER	ConCISE	DEER	ConCISE
Output Tok	11,307	8,669	8,996	7,242	10,752	5,915
Compute Tok	16,535	11,172	12,547	9,454	12,883	11,541
Ratio	1.46x	1.29x	1.39x	1.31x	1.20x	1.95x

DEER incurs additional overhead through repeated trial answer induction and confidence probing at each decision point, resulting in a consistent 1.20–1.46$\times$ gap across all model scales. ConCISE

exhibits a more pronounced and model-dependent overhead: while the gap is moderate for DeepSeek models (1.29–1.31×), it reaches 1.95× for Qwen3-4B, suggesting that the regeneration overhead is highly sensitive to the model's generation characteristics. In contrast, STARS performs no internal regeneration—suffix injection adds only a small number of tokens to the context while leaving the standard autoregressive decoding flow intact, thereby introducing no regeneration overhead.

J.2 Latency and throughput analysis

Latency and throughput are measured under a fixed evaluation protocol shared across all methods. Results are averaged across all three evaluated models on 100 problems sampled from a diverse mix of benchmarks including AIME, MathPerturb, and ReasoningTrap, covering a broad range of reasoning difficulty and generation length distributions. For each method, we report averages over multiple models and runs using identical hardware and decoding settings.

Table 15: **Latency and throughput comparison** (mean over models). Latency is measured as end-to-end runtime per example. $p50$ and $p95$ denote the median and 95th-percentile latency, respectively.

Method	Latency avg (s)	Latency p50 (s)	Latency p95 (s)	Throughput (tok/s)
Baseline	286.06	210.90	786.35	43.86
DEER	416.92	239.89	1356.40	35.72
ConCISE	278.58	210.99	724.91	34.34
STARS	**251.34**	**206.67**	**607.49**	**47.09**

Table 15 provides a detailed latency and throughput breakdown across inference-time baselines. DEER exhibits a substantially larger tail latency (p95), reflecting the input-dependent variability introduced by entropy-based early-exit decisions. In particular, reasoning trajectories that include frequent self-reflection or explicit uncertainty markers tend to maintain high token-level entropy, delaying confident early-exit decisions and increasing runtime variance. Because exit decisions are made dynamically based on intermediate confidence estimates, the amount of computation performed can vary widely across inputs, leading to long-tail latency behavior. While DEER proposes branch-parallel decoding to amortize the cost of repeated confidence probing, this mechanism requires maintaining multiple concurrent reasoning branches and incurs additional memory overhead. As a result, under single-trajectory decoding settings commonly used in practice, DEER can exhibit increased latency variance.

ConCISE reduces average latency in some cases by compressing low-utility intermediate steps, but incurs lower decoding throughput due to step-wise confidence evaluation and frequent intervention within the decoding loop. Because this control is applied at the token or step level, decoding sequences that contain frequent self-reflective or corrective tokens (e.g., explicit reconsideration or revision cues) can trigger repeated confidence checks and compression decisions, further increasing control overhead. This token-level control introduces additional synchronization and decision overhead, which can limit throughput despite reduced generation length.

It is important to note that the generation length reported for ConCISE reflects only the length of the final output trace, and does not account for the total number of tokens generated internally during decoding. Due to its confidence-guided regeneration mechanism, ConCISE may repeatedly invalidate and re-generate intermediate steps when reflective reasoning is detected. As a result, the internal decoding cost can substantially exceed the final generation length, despite the compact appearance of the output.

In contrast, STARS shows consistently lower tail latency and stable median performance, indicating that its inference-time behavior remains predictable across inputs. This stability arises from the sparse and event-driven nature of its intervention logic: flip and recurrence detection rely solely on lightweight hidden-state displacement signals and simple thresholding, and are triggered only at a small number of structurally significant points. Moreover, by selectively correcting unproductive reasoning trajectories rather than performing continuous token-level control, STARS often reduces unnecessary generation length while preserving the standard autoregressive decoding flow. As a result, STARS attains both lower end-to-end latency and higher decoding throughput than competing inference-time baselines.

These results suggest that the efficiency gains of STARS stem from stable and selective trajectory correction, rather than aggressive early termination or computationally intensive inference-time control mechanisms.

K QUALITATIVE CASE STUDIES

Inertia failure cases. The following examples contrast vanilla decoding and STARS on problems where the model reaches the correct answer but abandons it under parametric prior pressure. Vanilla decoding falls into **Cognitive Inertia**, characterized by prolonged hesitation, repeated self-correction, and late-stage commitment to incorrect interpretations. The model repeatedly signals uncertainty through phrases such as "Maybe I'm misinterpreting the problem", "Perhaps my initial assumption is wrong", or "That seems too straightforward", while continuing to elaborate on the same flawed hypothesis (Figure 10 and Figure 12). Rather than triggering early revision, these self-doubts accumulate into extended verification loops, eventually culminating in a confident but incorrect conclusion. By contrast, STARS intervenes at detected pivot points and steers the trajectory back toward a coherent interpretation. As illustrated in Figure 11 and Figure 13, the injected suffixes encourage a structural re-evaluation of the existing reasoning, allowing the model to abandon unstable interpretations before they become entrenched, recovering correct solutions with significantly reduced token consumption.

Plain failure cases. Beyond inertia-driven failures, vanilla decoding also fails on problems where the correct answer is never reached due to local reasoning errors. As shown in Figure 14, the model correctly determines that 20 cars are white, but then incorrectly treats the 8 two-door white cars as white four-door cars, leading to an incorrect answer of 13. As illustrated in Figure 15, STARS initially follows the same erroneous path, but a single intervention prompts the model to re-evaluate the stated constraints, correctly infer that 12 white cars are four-door, and recover the correct answer of 9. This example shows that STARS can also recover some Plain Failures even when the correct answer is never reached in the original reasoning trajectory.

Failure modes of STARS. However, STARS is not universally effective. We identify two recurring failure modes. First, STARS may fail when the reasoning process terminates before any salient latent spike is observed, leaving no opportunity for suffix injection or trajectory correction (Figure 16). Second, even when a spike is detected and an intervention is triggered, the injected suffix can occasionally steer the model toward an incorrect alternative trajectory (Figure 17). These cases highlight that while latent spikes often coincide with meaningful internal transitions, they do not guarantee correct redirection, underscoring that STARS functions as a selective mitigation mechanism rather than a universal safeguard against reasoning failure.

Example 1: Vanilla

Question.

For each positive integer n, let a_n be the least positive integer multiple of 23 such that $a_n \equiv 1 \pmod 7$. Find the number of positive integers $n \leq 1000$ such that $a_n = a_{n+1}$.

Generated Text.

...

Therefore, $k \equiv 4 \pmod 7$. So, the smallest positive integer k is 4. Therefore, $a_n = 23 \times 4 = 92$. Wait, so is a_n always 92? ... If a_n is always 92, then $a_n = a_{n+1}$ for all n, so the number of such n would just be 1000. **But that seems too straightforward.**

...

Maybe I'm misinterpreting the problem. Wait, perhaps a_n is defined for each n... **Alternatively, perhaps it's the least multiple of 23 greater than or equal to n that is congruent to 1 mod 7.** That interpretation could make more sense, as it would vary with n. Let me try that interpretation. ... So, the sequence a_n is 92, 253, 414, 575, 736, 897, 1058...

...

Thus, total is 902. Therefore, the number of positive integers $n \leq 1000$ such that $a_n = a_{n+1}$ is 902.

Final Answer. $\boxed{902}$ (9067 tokens)

Figure 10: Baseline failure on example 1 (DeepSeek-7B on ConditionedMath).

Example 1: STARS

Question.

For each positive integer n, let a_n be the least positive integer multiple of 23 such that $a_n \equiv 1 \pmod 7$. Find the number of positive integers $n \leq 1000$ such that $a_n = a_{n+1}$.

Generated Text.

...

But that seems too straightforward, so maybe I need to check again.

Wait. I am shifting my reasoning. Let's double-check if this direction is valid and grounded in the text.

The problem says: For each positive integer n, let a_n be the least positive integer multiple of 23 such that $a_n \equiv 1 \pmod 7$. So, each a_n is a multiple of 23 such that $a_n \equiv 1 \pmod 7$.

...

But in the problem, it's written as $a_n \equiv 1 \pmod 7$, without n, so it's fixed. Therefore, yeah, 92 is the same for all n.

Wait. This line of thinking is looping. Let's pause and pivot. Is there a simpler way to look at this problem?

Maybe considering the definition of a_n. Each a_n is the minimal multiple of 23 that is 1 mod 7. Since the minimal solution is 92, which is fixed, so all $a_n = 92$. Therefore, indeed, $a_n = a_{n+1}$ for all n, so every n from 1 to 1000 satisfies it. So, the answer is 1000.

Final Answer. $\boxed{1000}$ (4134 tokens)

Figure 11: STARS success on example 1 (DeepSeek-7B on ConditionedMath).

Example 2: Vanilla

Question.

There are $8! = 40320$ eight-digit positive integers that use each of the digits $1, 2, 3, 4, 5, 6, 7, 8$ exactly once. Let N be the number of these integers that are divisible by 22. Find the difference between N and 2025.

Generated Text.

...

But in each case, the sum is too high or too low. **Wait, that can't be. So, perhaps my initial assumption is wrong.** ... Therefore, in all cases, it's impossible to have sum of odd digit in even positions $= 2$, since the smallest odd digit is 1. Therefore, there are no such numbers, so $N = 0$. **But earlier, the combinatorial approach suggested $N = 128$. So, which is correct?**

...

Therefore, $N = 0$. Hence, $N - 2025 = -2025$. But perhaps the problem expects the absolute difference, which is 2025. But since the problem says "difference", which is a signed number, so it's -2025. But I think, given the analysis, the correct answer is $N = 0$, so $N - 2025 = -2025$.

Final Answer. $\boxed{-2025}$ (18104 tokens)

Figure 12: Baseline failure on example 2 (DeepSeek-7B on AIME 2025).

Example 2: STARS

Question.

There are $8! = 40320$ eight-digit positive integers that use each of the digits $1, 2, 3, 4, 5, 6, 7, 8$ exactly once. Let N be the number of these integers that are divisible by 22. Find the difference between N and 2025.

Generated Text.

...

Total subsets: $4 + 3 + 1 = 8$ subsets. So, there are 8 subsets S of $\{1, 2, 3, 4, 5, 6, 7, 8\}$ with 4 elements that sum to 18.

Wait. I am shifting my reasoning. Let's double-check if this direction is valid and grounded in the text.

So, if each subset S corresponds to a unique set of digits in the even positions, which must include two even digits.

...

Therefore, for each subset S, the number of valid numbers is $2 \cdot 3! \cdot 4! = 288$.

Wait. This line of thinking is looping. Let's pause and pivot. Is there a simpler way to look at this problem?

Maybe I can approach it differently. Wait, let me consider a smaller case to see.

...

Therefore, 2304 is indeed the correct count. Hence, the difference $N - 2025 = 2304 - 2025 = 279$.

Final Answer. $\boxed{279}$ (10240 tokens)

Figure 13: STARS success on example 2 (DeepSeek-7B on AIME 2025).

Plain Failure Example: Vanilla

Question.

There are 36 cars in my building's parking lot. All of the cars are red, white or blue, and a car can have either 2 doors or 4 doors. 10 of them are red, 21 of them are 4-door, 6 of them are blue, and 8 of them are 2-door and white. How many of the cars are 4-door and but not white?

Generated Text.
There are 36 cars in total. Out of these, 10 are red, 6 are blue, and the remaining 20 must be white.
Next, I'll look at the number of 4-door cars. There are 21 4-door cars in total. Since 8 of these 4-door cars are white, the remaining 13 4-door cars must be either red or blue.

...

Thus, the number of 4-door cars that are not white is

$$21 - 8 = 13.$$

Final Answer. $\boxed{13}$ (384 tokens)

Figure 14: Baseline plain failure (DeepSeek-1.5B on MathPerturb).

Plain Failure Example: STARS

Question.

There are 36 cars in my building's parking lot. All of the cars are red, white or blue, and a car can have either 2 doors or 4 doors. 10 of them are red, 21 of them are 4-door, 6 of them are blue, and 8 of them are 2-door and white. How many of the cars are 4-door and but not white?

Generated Text.
There are 10 red cars, 21 four-door cars, 6 blue cars, and 8 cars that are both two-door and white. I want to find the number of four-door cars that are not white.
There are 21 four-door cars in total. To find out how many of these are not white, I subtract the number of white four-door cars: $21 - 8 = 13$.

Wait. I am shifting my reasoning. Let's double-check if this direction is valid and grounded in the text.

...

Step 1: Determine the number of White cars. White $= 36 - 10 - 6 = 20$.
Step 2: Determine the number of 4-door White cars. Since 8 White cars are 2-door, 4-door White $= 20 - 8 = 12$.
Step 3: Subtract from the total 4-door cars. 4-door not White $= 21 - 12 = 9$.

Final Answer. $\boxed{9}$ (675 tokens)

Figure 15: STARS recovery of a plain failure (DeepSeek-1.5B on MathPerturb).

Failure Mode I: Termination before Suffix Injection

Question.

Sam is carrying a 2 gallon bucket of milk to his house from the barn, 3 miles away. The bucket leaks so that the amount of milk remaining is $(\frac{2}{3})^n$ gallons after n miles. How many gallons of milk will be in the bucket when Sam gets home?

Generated Text.

...

After each mile, it's multiplying by $2/3$. So, after 1 mile: $2 \cdot (2/3) = 4/3$, but the formula says $(2/3)^1$ is $2/3$. Wait, now I'm confused.
Wait, let's test $n = 0$. After 0 miles, it's $(2/3)^0 = 1$ gallon. Wait, but he starts with 2 gallons. That doesn't add up. Therefore, perhaps the formula is missing a scaling factor.

Therefore, perhaps the amount is $2 \cdot (2/3)^n$.

...

Therefore, $M(n) = 2 \cdot (2/3)^n$. So, $M(3) = 2 \cdot (8/27) = 16/27$.
Alternatively, if the formula is just $(2/3)^n$, that doesn't account for the initial 2 gallons, so I think that approach is wrong.

Final Answer. $\boxed{\frac{16}{27}}$ (5112 tokens)

Figure 16: **Failure mode I: Termination before suffix injection** (DeepSeek-7B on Conditioned-Math).

Failure Mode II: Misguided Suffix Steering

Question.

Among the 900 residents of Aimeville, there are 195 who own a diamond ring, 367 who own a set of golf clubs, and 562 who own a garden spade. In addition, each of the 900 residents owns a bag of candy hearts. Suppose the 234 residents who own exactly three of these things also own a candy store, which means all four items. Find the number of residents who own all four of these things.

Generated Text.

...

In general, for each triplet of sets, the number of people in exactly that triplet is $|A \cap B \cap C| - |A \cap B \cap C \cap D|$.

Wait. I am shifting my reasoning. Let's double-check if this direction is valid and grounded in the text.

So, since the problem says that 234 residents own exactly three of these things and also own a candy store, meaning all four. So, I think it's best to assume that these 234 residents actually own all four items. Therefore, the number of residents who own all four is 234.

...

But $0 = 672$? That can't be. So, contradiction. *Hmm, that suggests that there's an inconsistency in the assumptions.*

...

Since $xyz = 234$, $10 + 2 \cdot 234 = 478$, which matches $|\text{only } D| = 478$. So, all numbers check out.
Therefore, the number of residents who own all four is $xyz = 234$.

Final Answer. $\boxed{234}$ (11225 tokens)

Figure 17: **Failure mode II: Misguided suffix steering** (DeepSeek-7B on ConditionedMath).